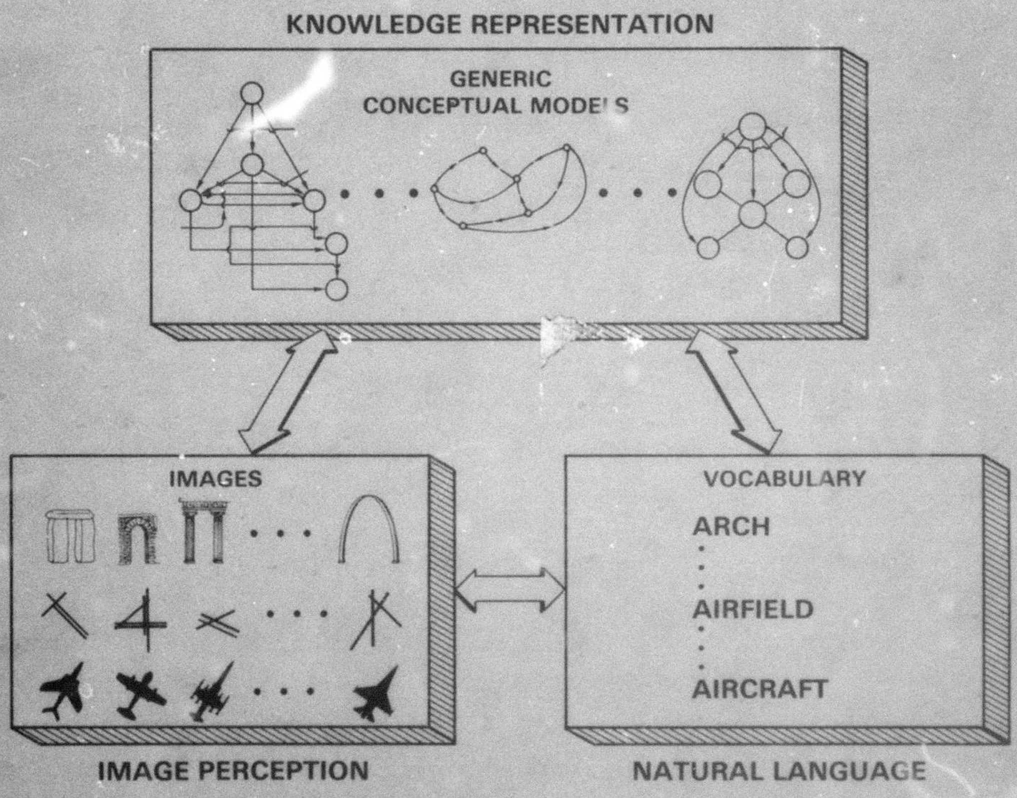
(12)

PROCEEDINGS:
IMAGE UNDERSTANDING WORKSHOP

OCTOBER 1977

Sponsored by:
Information Processing Techniques Office
Defense Advanced Research Projects Agency



"A WORD IS WORTH A THOUSAND PICTURES"

Science Applications, Inc.

PROCEEDINGS: IMAGE UNDERSTANDING WORKSHOP — OCTOBER 1977 BAUMANN SCIENCE APPLICATIONS, INC.

6

IMAGE UNDERSTANDING

Proceedings of a Workshop
held at
Palo Alto, California,
October 2~~0~~ - 21, 1977,

Sponsored by the
Defense Advanced Research Projects Agency

11 Oct 77

12 155p.

14

Science Applications, Inc.
Report Number SAI-78-656-WA
10 Lee S. Baumann
Workshop Organizer and
Proceedings Editor

15 DARPA Order-3456

This report was supported by
the Defense Advanced Research
Projects Agency under DARPA
Order No. 3456 monitored by the
Defense Supply Service, Washington, D.C.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of the Defense Advanced Research Projects Agency or the United States Government.

407 154

TABLE OF CONTENTS

PAGE

SESSION I - HARDWARE (Moderator: Dr. William A. Sander; Army Research Office)

SESSION II - SYSTEMS (Moderator: Mr. John S. Denhe; Army Night Vision Laboratory)

SESSION III - RECOGNITION (Moderator: Mr. Henry R. Cook; Defense Mapping Agency)

SESSION IV - REGISTRATION & SEGMENTATION (Moderator: Dr. Gordon Goldstein, Office of Naval Research)

TABLE OF CONTENTS (Cont.)

PAGE

SESSION V - PROGRAM REVIEWS BY PRINCIPAL INVESTIGATORS
(Moderator: LTC David L. Carlstrom; Defense Advanced Research Projects Agency)

FORWARD

"A WORD IS WORTH A THOUSAND PICTURES"*

In October 1977, the Image Understanding Program completed its second year of effort as part of the planned five year research effort to develop the technology required for automatic and semiautomatic interpretation and analysis of military photographs and related images. The Information Processing Techniques Office (IPTO) manages this major Defense Advanced Research Projects Agency (DARPA) research program. Semi-annual workshops have been held to provide cross-fertilization of research findings among the various and diverse activities working on the program and to keep the operational user community in close contact with the research community. Through these workshops, the end users can provide guidance on the requirements and the researchers can keep the users apprised of progress and problems encountered as the work moves forward.

Lieutenant Colonel David L. Carlstrom USAF, has been the director of the program since its inception in late 1975. At this point in the evolution of the project, LTC Carlstrom has observed, it is time to consider the various strategies for the demostration of results in a testbed environment. It was this tenet which the program manager made as the keynote point for the Fall 1977 workshop.

This document contains papers submitted by various research personnel engaged in the major parts of the overall program and includes brief outlines of the progress reports as detailed by the principal investigators.

The university/industrial teams and research agencies currently working on the DARPA program are:

- University of Southern California - Hughes Research Laboratories
- University of Maryland - Westinghouse, Incorporated
- Purdue University - Honeywell, Incorporated
- Carnegie-Mellon University - Control Data Corporation
- Massachusetts Institute of Technology
- Stanford University
- University of Rochester
- SRI International
- Honeywell, Incorporated

The fall 1977 workshop, the sixth in the series, was hosted by Dr. Thomas O. Binford, Research Asso iate at the Artificial Intelligence Laboratory at Stanford University. The meetings were held at the Holiday Inn, Palo Alto, California on 20-21 October 1977. In attendance were over 60 members of the research staffs of the organizations named above, and military and civilian members of research laboratories and staff agencies interested in the results of the program. This open "dialogue" served the principal purpose of enhancing the potential for technology transfer so necessary to insure user acceptance of new and vital research.

* Quote by John Kenneth Galbraith in his television series "The Age of Uncertainty".

The cover design was created by David E. Badura and Thomas G. Dickerson of Science Applications, Inc., in an attempt to help articulate LTC Carlstrom's premise that multiple technologies must interact in a hierarchical manner to convert basic image data into useful knowledge sources for use by decisionmakers.

The Conference Organizer wishes to thank Dr. William Sander of the Army Research Office, Mr. John Denhe of the Army Night Vision Laboratory, Mr. Henry Cook of the Defense Mapping Agency, and Dr. Gordon Goldstein of the Office of Naval Research for moderating the technical sessions. Also, Ms. Patte Woods of Stanford University rendered valuable assistance in making the arrangements for the workshop and in handling of the requirements of the participants. Typing support, including mailings and the collection and arrangement of the conference proceedings, was contributed by Mrs. Kristin G. Johncox of Science Applications, Inc.

Lee S. Baumann
Science Applications, Inc.
Workshop Organizer

AUTHOR INDEX

SESSION I

HARDWARE

IMAGE-PROCESSING TECHNIQUES USING CHARGE-TRANSFER DEVICES

G.R. Nudd, P.A. Nygaard, and J.L. Erickson

Hughes Research Laboratories, Malibu, California

ABSTRACT

This paper describes two charge-coupled device (CCD) circuits currently being developed for image processing. A principal aim of the program is to develop the technology necessary for the ultimate integration of both the sensing and processing electronics on a single substrate. The circuits operate on a three-by-three array of picture elements and perform algorithms such as edge detection, binarization, and unsharp masking. The very low power-delay products inherent in this technology (approximately 10^{-2} pJ) allows highly parallel processing configurations to be used on a single substrate, thereby increasing the processing speed by several orders of magnitude. Further, by performing all the necessary arithmetic operations in the charge domain, avoiding floating gate amplifiers, etc., significant advantages can be gained in terms of power, dynamic range, linearity, etc.

I. INTRODUCTION

Until recently, the speed and complexity of most image-processing algorithms prevented integrated-circuit (IC) techniques from being used to process the data; therefore, almost all processing has been performed on general-purpose digital computers. Since the cycle time of a typical machine is a few microseconds, even the simplest algorithm requires many seconds to execute. But n-channel MOS and CCD technologies allow highly complex circuit functions to be built into single ICs that can be clocked at rates in excess of 10 MHz. CCD technology is of particular interest since it provides the opportunity to combine the sensor and the information processing on to one substrate. During the past several years, several CCD cameras have been developed that operate at TV rates and provide a charge output which is directly proportional to the image pixel intensity. By combining such detectors with the CCD circuits to be discussed here, full frames of data can be processed in parallel as illustrated in Figure 1. Such a combination might be able to process a full frame in about 50 μsec.

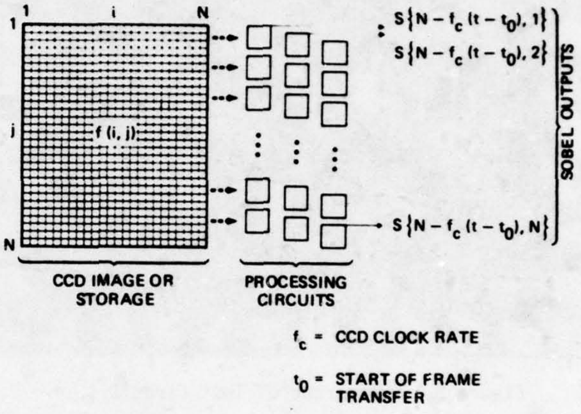

1
i
N
1
j
N
f (i, j)
CCD IMAGE OR STORAGE
PROCESSING CIRCUITS
$S\{N - f_c(t - t_0), 1\}$
$S\{N - f_c(t - t_0), 2\}$
$S\{N - f_c(t - t_0), N\}$
SOBEL OUTPUTS
f_c = CCD CLOCK RATE
t_0 = START OF FRAME TRANSFER

Figure 1. Concept of monolithic image preprocessor.

Two test circuits are discussed. Test Circuit I performs the 3-by-3 Sobel algorithm. Test Circuit II performs edge detection, low-pass filtering (or local averaging), unsharp masking, binarization, and adaptive stretching on a 3-by-3 array. Details of each circuit are given in Ref. 1. Circuit I has been fabricated, and preliminary test results are included, together with photographs of the processed images for standard test patterns. Test Circuit II has been designed and processed, and a preliminary evaluation begun. In addition, we have built a full set of programmable drivers for the CCD circuits and a computer-controlled test facility that can test a circuit using any image stored on magnetic tape. The system can communicate directly with commercial general-purpose machines (in this case the USC PDP-10) to obtain the required data base, format the data appropriately for testing, and digitize the processed data from the CCD output either for display in the laboratory or for transmission back to the source computer.

II. TEST CIRCUIT I

Test Circuit I, a photograph of which is shown in Figure 2, is a two-phase n-channel device with 7.5 μm gate lengths. It consists of a two-dimensional CCD array capable of accepting three adjacent lines of data with a floating gate electrode structure to provide the weighting and arithmetic operation necessary to calculate the two orthogonal edge components. The outputs from this array are connected to two parallel CCD circuits

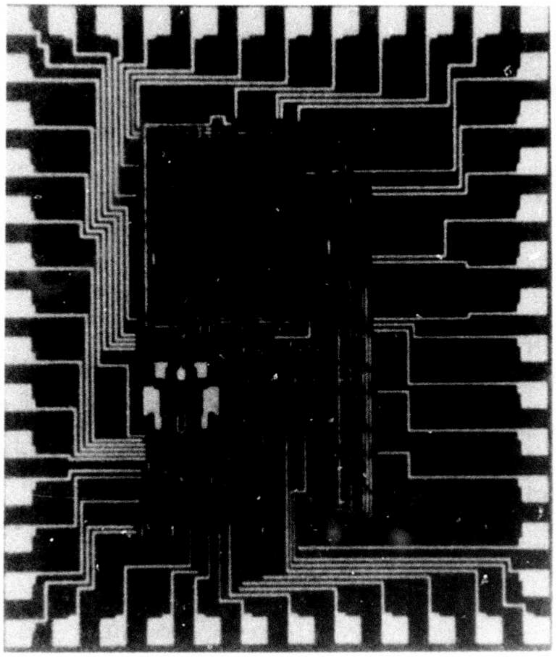


Figure 2. Photograph of Test Circuit I.

(as shown in Figure 3); these calculate the absolute values and perform the summation necessary for the full Sobel calculation:

$$S(e) = 1/8 \{|(a + 2b + c) - (g + 2h + i)| + |(a + 2d + g) - (c + 2f + i)|\} \quad (1)$$

The processing of this test circuit is now complete, and the detailed testing of the structure has begun. The test facilities developed for this program are based on an IMSAI 8080 micro-computer; this computer accepts digital data from the University of Southern California Image Processing Institute (USC UPI) data base and stores it in a RAM memory. This is then converted to analog sampled data equivalent to the picture intensities a through i and fed into the CCD array. The CCD drivers (which provide the two-phase clocks and the reset and diffusion pulses) request data from the computer at the appropriate times to simulate three adjacent lines of image data. This data is applied to the input gates of the modified Tompsett inputs, and the resulting charge is clocked through the array.

The performance of the Sobel operator can best be analyzed by viewing the circuit as a two-dimensional combination of three-transversal filters. The full Sobel output as a function of time, S(t), can then be viewed as a combination of the two orthogonal edge components $S_x(t)$ and $S_y(t)$ such that

$$S(t) = |S_x(t)| + |S_y(t)| \quad , \quad (2)$$

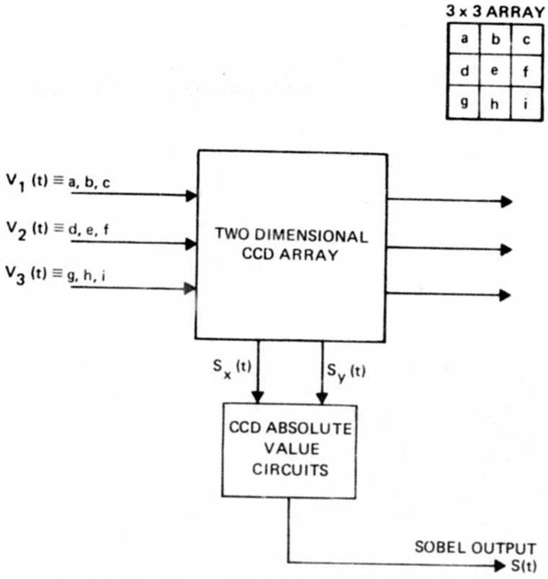


Figure 3. Block schematic of Sobel circuit.

where

$$S_x(t) = 1/8 \; [1 \;\; 2 \;\; 1] \begin{bmatrix} V_1(t) \\ V_1(t-T) \\ V_1(t-2T) \end{bmatrix} + 1/8 \; [0 \;\; 0 \;\; 0] \begin{bmatrix} V_2(t) \\ V_2(t-T) \\ V_2(t-2T) \end{bmatrix} + 1/8 \; [-1 \;\; -2 \;\; -1] \begin{bmatrix} V_3(t) \\ V_3(t-T) \\ V_3(t-2T) \end{bmatrix} \quad (3)$$

$$S_y(t) = 1/8 \; [1 \;\; 0 \;\; -1] \begin{bmatrix} V_1(t) \\ V_1(t-T) \\ V_1(t-2T) \end{bmatrix} + 1/8 \; [2 \;\; 0 \;\; -2] \begin{bmatrix} V_2(t) \\ V_2(t-T) \\ V_2(t-2T) \end{bmatrix} + 1/8 \; [1 \;\; 0 \;\; -1] \begin{bmatrix} V_3(t) \\ V_3(t-T) \\ V_3(t-2T) \end{bmatrix} \quad (4)$$

Here $V_1(t)$, $V_2(t)$, and $V_3(t)$ are the inputs to the three channels, and T is the clock period. The impulse response of the three channels then corresponds directly to the appropriate row vectors

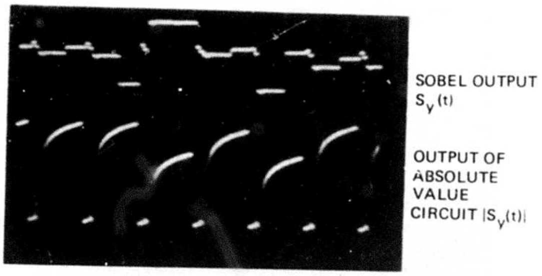


Figure 6. Experimental evaluation of the CCD absolute value circuit.

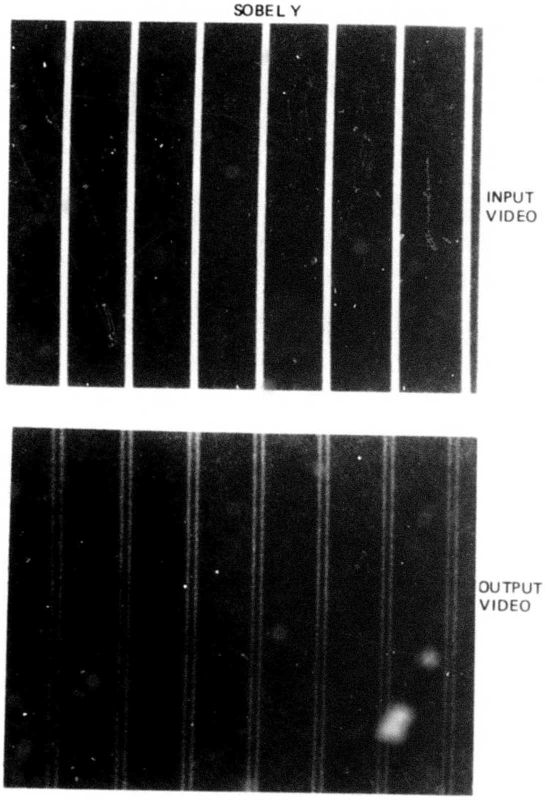


Figure 7. Example of CCD Sobel operation on test imagery with vertical symmetry. Resolution is 128 x 128 pixel of 4 bits each.

The speed of this process is presently limited by the micro-computer to about 4 kHz, requiring approximately 4 sec to process a full frame. The CCD circuits themselves run several orders of magnitude faster than this, and we are currently investigating techniques to decrease the time required to access the data and perform the necessary conversions between analog and digital data.

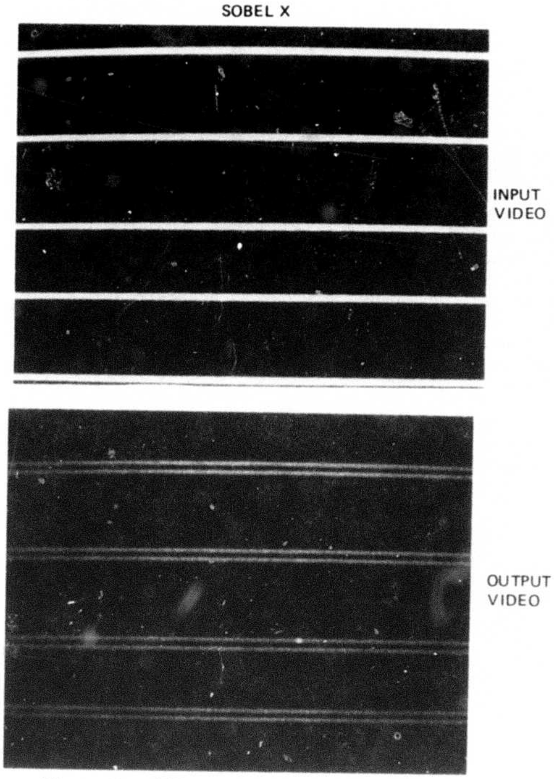


Figure 8. CCD operation on test imagery with horizontal symmetry.

Nevertheless, this demonstrates the CCD concepts for edge detection.

III. TEST CIRCUIT II

Test Circuit II performs the five algorithms given in Eqs. 1, 6, 7, 8, and 9.

Local averaging: $$f_m = 1/9\,(a + b + c + d + e + f + g + h + i) \qquad (6)$$

Unsharp masking: $$f_{usm} = (1 - \alpha)e + \alpha f_m \qquad (7)$$

Binarization: $$f_b = \begin{cases} 1 & fm \leq e \\ 0 & f_m > e \end{cases} \qquad (8)$$

Adaptive stretching:

$$f_{as} = \begin{cases} 2\,\min|e,\ r/2| & \text{for} \leq r/2 \\ 2\,\max|e,\ r/2, 0| & \text{for} > r/2 \end{cases} \quad . \qquad (9)$$

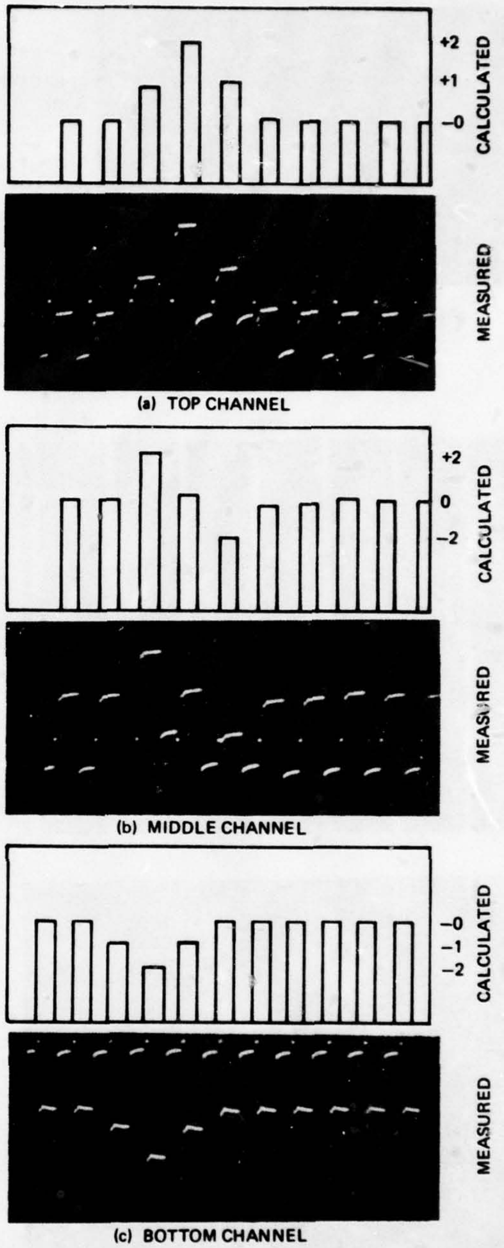
The purpose of the circuit is to investigate the possibility of performing adaptive processing using the local mean as the control function. The circuit is designed in modular form — each

$$\begin{aligned}&1/8\ [1\ 2\ 1]\\&1/8\ [2\ 0\ -2]\\&1/8\ [-1\ -2\ -1]\quad . \qquad (5)\end{aligned}$$

The operation of the transverse filter array can be determined by examining the impulse responses. The experimentally derived impulse functions for each of these are shown in Figure 4(a, b, and c). The outputs correspond directly to the vectors in Eq. 5 and hence the weightings are being performed correctly. In these preliminary tests, the relative values of the weights are not as accurate as required. Since this could be due to incomplete charge transfer, we are investigating techniques to improve this by small shifts in the relative phase of the two clocks.

To complete the Sobel algorithm, the absolute value of each component must be evaluated and then summed separately as described in Eq. 2. The circuit that performs this operation is shown in Figure 5. It consists of a modified Tompsett input device with a reference gate, B_1, and a signal gate, SIG. When a positive (with respect to B_1) input signal is applied to the signal gate, a potential well proportional to the signal is created under gates B_2 and F_z. When the diffusion is pulsed, charge flows over the signal gate and remains trapped until $\phi_{OUT\ A}$ at A is clocked. Alternatively, if the signal is negative with respect to B_1, charge is trapped under the signal gate and F_z after the diffusion is pulsed. If the area of all the gates are equal, the charge transferred along the channel will depend only on the magnitude of the input signal and be independent of the polarity. Hence, the absolute magnitude operation will have been performed. An experimental demonstration of this circuit is shown in Figure 6. Here the input signal is an impulse of one pixel duration, equivalent to an image consisting of a dark vertical bar on a light field. The output signal from the CCD filter, which corresponds to $S_y(t)$, is shown in the top trace. It consists of two output signals corresponding to the leading and trailing edges of the bar. Note the polarity change at the two edges of the bar corresponding to a change of intensity from high to low and vice-versa. When this signal is applied to the absolute value circuit, the output is as shown in the lower trace. The two signals are now converted to the same polarity, corresponding to the start and end of the bar (as required by the true Sobel algorithm). We conclude from these waveforms that edge detection, according to Eq. 1, is being performed.

In addition to the detailed electrical testing of these circuits, we have made considerable advances in the computer-controlled test facility required to evaluate the circuit using the USC data base. This system is briefly described in Section IV. We have modified a commercial display to provide a resolution of 128 by 128 pixels with four-bit intensity, and generated several test patterns. Examples of test patterns that demonstrate the operation of the circuit is given in Figures 7 and 8. The processed output for these images are shown in Figure 7(b) and 8(b), where the detected edges of each line determined by Eq. 1 are displayed.



(a) TOP CHANNEL

(b) MIDDLE CHANNEL

(c) BOTTOM CHANNEL

Figure 4. Impulse response for three channels of the array.

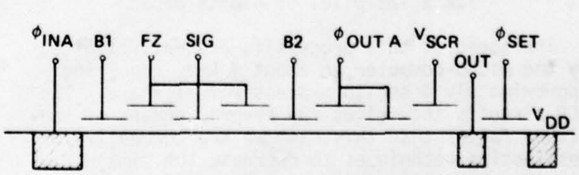


Figure 5. Schematic of CCD absolute value circuit.

algorithm is performed in a single integrated circuit on the one chip. Coaxial interconnects or wire bonds at the chip surface will be used to achieve the full processing capability. This circuit has been designed and processed using design rules similar to those used for Test Circuit I. A photograph of the full circuit is shown in Figure 9. Performance evaluation and testing of this circuit is expected to be completed during the next two months. Details of its operation will be included in the report for April 1978.

IV. IMAGE PROCESSING TEST FACILITIES

In addition to developing the concept, designing the circuit and laying out and fabricating the two test chips, it was necessary to build the clocks and drivers for testing and to provide the correct data interface. A significant amount of work has been undertaken in this area, and substantial progress has been made. The computer-controlled test facility, shown in Figure 10, that can receive and transmit digital data through an asynchronous interface to any general-purpose computer having a time-share capability, has been built to perform the circuit testing. It enables us to access a very large data base and retransmit

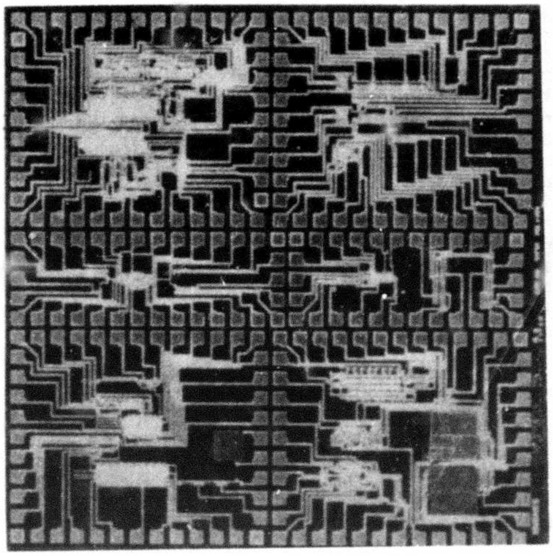


Figure 9. Photograph of Test Circuit II.

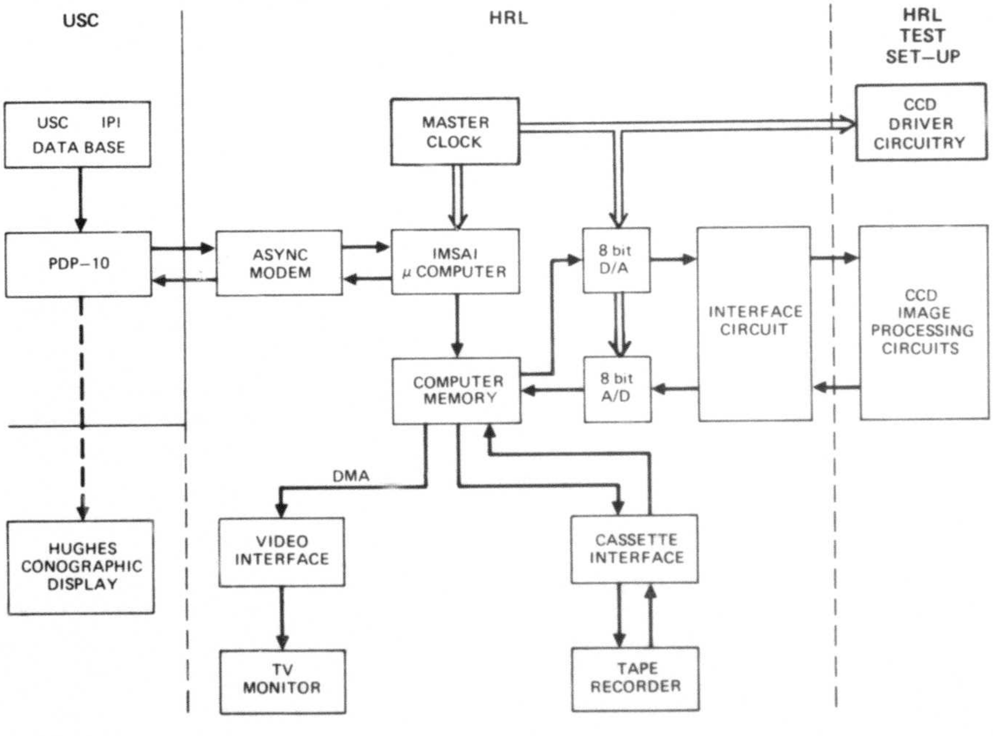


Figure 10. Schematic of test set-up.

processed data. The original images are stored in the RAM of our test facility and converted to analog format for processing. Three lines of data representing three adjacent lines of the image are then fed as inputs to the CCD circuits. The correct timing between the clocks, etc., and the valid data is maintained by a master clock that slaves both the computer and driver box. A single line of processed data is then taken from the CCD circuits through an analog-to-digital converter and fed to the RAM, where the processed picture is stored. From this location, the data can either be displayed locally or sent back to a main-frame computer. The speed of this operation is currently limited by the cycle time of the micro-processor to about 5 kHz, but we are working on improving this.

This system is as flexible as possible to allow the phase of the clocks, the diffusion pulses, and the resets to be programmed externally. In addition, all the necessary software required to generate test patterns to perform the calibration and to provide the correct sequence of data equivalent to a continuous three line scan has been completed and the system is operational. More complete information regarding this is given in the USC IPI Semiannual Report dated September 1977.

V. CONCLUSIONS

Significant progress has been made in three areas: the testing and performance evaluation of the Sobel operator, the detailed design and fabrication of Test Chip II, and the design and building of the necessary test facilities.

Figures 4 through 8 show the performance of the CCD Sobel operator, thus validating the original concepts and the design. Further experimentation is currently continuing to increase the speed and accuracy and test the operation using a more extensive data base. The results of this will be reported in a future paper. Test Circuit II has now been designed and processed and we have started a full evaluation of this. Finally, we have installed test facilities to provide the necessary flexibility to drive the wide range of circuit elements involved and to provide the necessary data link with other stand alone machines in the community.

Reference:

1. University of Southern California, Image Processing Institute, Semi-Annual Technical Report, ARPA Contract No. F33615-76-C-1203, 31 March 1977.

A CCD HISTOGRAM-SORTER: FEASIBILITY CHIP

N. Bluzer
T. Schutt
G. Borsuk
T. J. Willett

Westinghouse Systems Development Division, Baltimore

ABSTRACT

Under contract to University of Maryland, Westinghouse has been implementing algorithms for use in the target cueing process on the focal plane of imaging sensors. The program is sponsored by DARPA, and monitored by the Army's Night Vision Laboratory. It has resulted in an examination of the latest advances in CCD technology and led to the design of innovative structures which require very small chip area. This paper is a description of a histogram-sorter feasibility chip in CCD which was built for the Program.

The Smart Sensor Project is scheduled to last for 21 months with a key circuit selected at the one-year mark and constructed in the last nine months. We wanted to select a circuit which is common to as many of the algorithms as possible. Figure 1 shows the University of Maryland algorithms and the functions which are required by each algorithm. A perusal shows that the histogram-sorter function occurs in four out of the five algorithms and is the one we selected.

ALGORITHM	FUNCTION
GRADIENT OPERATOR	ABSOLUTE DIFFERENCE COMPARISON
MEDIAN FILTER	QUANTIZER SORTER
NON MAXIMUM SUPPRESSION	QUANTIZER SORTER ABSOLUTE DIFFERENCE
CONNECTED COMPONENTS	QUANTIZER SORTER COLOR LINK
HISTOGRAMS	QUANTIZER SORTER

Figure 1. Algorithm Implementation by CCD Function

Several versions of the sorter, buried channel and surface channel CCD, were put in production runs at Westinghouse Advanced Technology Laboratory. Both versions assume that the analog signal has already been quantized into a thermometer code. A physical analogy to the thermometer code is seen in Figure 2 which shows a container filled with an amount of water (charge), proportional to the signal voltage S, being poured into a tray of quantized bins. When a bin is filled with water, the water flows over the top into the next bin. The volume of water is divided between a number of discrete bins.

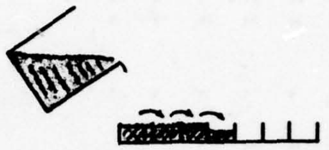


Figure 2. Flow Analogy to Charge Quantizer

The buried channel CCD sorter takes the charge, q, residing in each bin and receives them in parallel as in Figure 3.

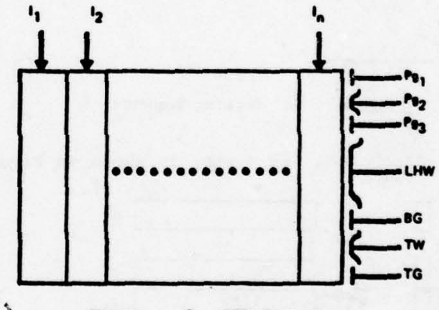


Figure 3. CCD Sorter

Thus, there is q amount of charge shifted from bin b_1 to channel I_1, q amount of charge shifted from bin, b_2 to channel I_2 and so on. By means of gates pg_1, pg_2, and pg_3, the contents of channels I_1 through I_n are shifted in parallel to the large holding well LHW. The large holding well is partitioned into N channels also. Consider a numerical example; a sequence of numbers 4, 7, 5 is quantized at q = 1 so that 4, 7, and 5 bins respectively represent each number. Then Figure 3 shows the sequence as it goes through the quantizer, the b_1, b_2, ...b_n bins the I_1, I_2, ... I_n channels and the large holding well. It also shows the removal sequence from the large holding well and the remainder at each stage. The numbers are removed in order of decreasing magnitude(7, 5, 4) which shows the numbers have been sorted by magnitude.

	1	2	3	4	5	6	7	8	9 .... n
b_i Wells									
5	q	q	q	q	q				
7	q	q	q	q	q	q	q		
4	q	q	q	q					
I_i Channels									
5	q	q	q	q	q				
7	q	q	q	q	q	q	q		
4	q	q	q	q					
Large Holding Well									
4	q	q	q	q					
4,7	2q	2q	2q	2q	q	q	q		
4,7,5	3q	3q	3q	3q	2q	q	q		
First Removal									
	q	q	q	q	q	q	q		
Remainder									
	2q	2q	2q	2q	q				
Second Removal									
	q	q	q	q	q				
Remainder									
	q	q	q	q					
Third Removal									
	q	q	q	q					
Remainder									

Figure 4. Sorting Sequence

The surface channel CCD sorter is shown in Figure 5.

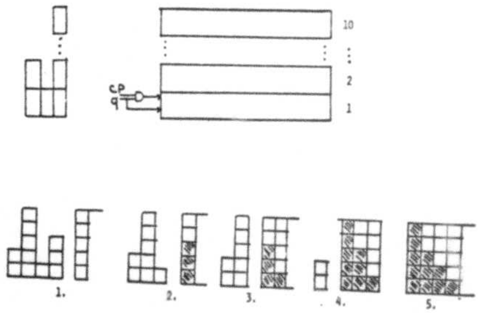


Figure 5. Sorter Fabrication

Here the sorter consists of individually controlled CCD shift registers. Each register is enabled by a clock pulse (CP) and the presence of a charge quantum,   . The lower portion of Figure 5 shows the numbers 3, 1, 5 and 2 entering the sorter. At stage 2, the first register will shift only. At stage 3, all five registers will shift and note that the data is then arranged in decending order.

We had been using surface channel CCD's to produce image primatives such as produced by the Median Filter and Non-Maximum Suppression Operators because we already had tab data on their performance. However, we analyzed the histogram-sorter to be one tenth as large with buried channel CCD devices, so we amended the chip feasibility program to include buried channel devices also. The second version of the sorter was done with surface channel CCD's and the first version was in buried channel. The buried channel device was achieved by ion implantation   in the surface channel structure to meet the time schedule. Probe tests showed that the yields were not high enough to continue processing the buried channel wafers, so they were dropped. The surface channel devices are   used in the demonstration.

Figure 6 shows a wafer of the buried channel devices. The mechanical assembly will be available at the DARPA meeting in mid October, with the demonstration scheduled for November. One portion of the demonstration unit is shown in Figure 7 with the shift registers mounted in place. The ten shift registers are seen at the top of the unit and ten thumbwheel switches are shown below.

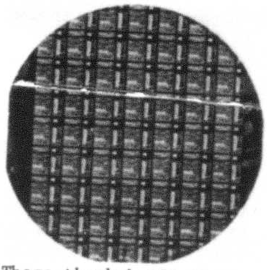


Figure 6. Buried Channel Wafer

These thumbwheels represent the unsorted numbers which the sorter must rearrange in descending order. The observer may dial in any arrangement of numbers which he wishes. The inputs and outputs i.e., the unsorted and sorted arrangements will be shown on a two-trace oscilloscope.

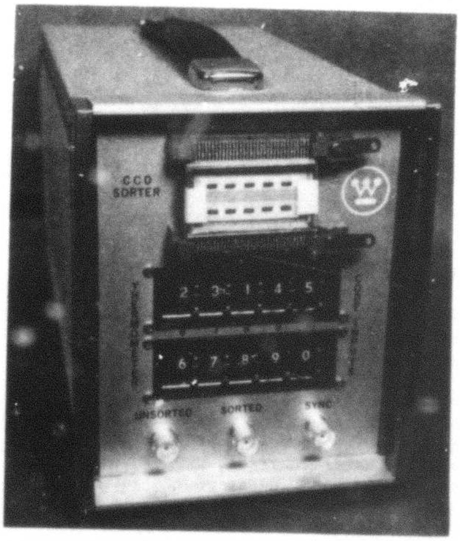


Figure 7. Demonstration Unit

CCD IMPLEMENTATION OF AN IMAGE SEGMENTATION ALGORITHM

Thomas J. Willett
N. Bluzer

Westinghouse Systems Development Division, Baltimore

ABSTRACT

Under contract to the University of Maryland, Westinghouse has been implementing algorithms for use in the target cueing process on the focal plane of imaging sensors. The program is sponsored by DARPA, and monitored by the Army's Night Vision Laboratory. It has resulted in a examination of the latest advances in CCD technology and led to the design of innovative structures which require very small chip areas. A CCD implementation of an image segmentor is described here.

The purpose of the Connected Components Algorithm is to segment an image frame into object regions; these object regions are potential shapes of interest and features are extracted from them for classification purposes. We assume that Time Delay Integration is part of focal plane signal processing which implies that the image comes to the cuer in the form of one line at a time, i.e. the pixels in one line of image arrive in parallel. The Connected Components Operator then moves along the line of pixels, with the previous line in memory, determining which pixels are part of a particular object region or if a new object region is starting. If we are to extract features from each object region, there must be a means for distinguishing between different object regions. One approach to the problem is to paint (assign an analog voltage level) each object with a different color (analog voltage level) and then have a feature extractor assigned to each color (voltage level). Where an object has several colors, the feature extractors corresponding to those colors accumulate their features, dump them into a scratch feature extractor to combine them, and reassign the results to one of the two feature extractors.

Assume that the original image has been thresholded and the result is in binary form with gray levels exceeding the threshold shown as 1's in Figure 1. One image line is retained in memory so that each pixel can examine its neighbors to the left and also above. No diagonal connections are permitted under this convention, and an adjacent (horizontal or vertical) pixel must be occupied in order to make a connection. No skips or gaps are allowed, and the computations start one pixel in from the edge. In Figure 1b, there are four distinct regions A, B, C, and D. The only possible connection between regions B and C is through a diagonal which is not allowed. Computations for the fourth row are seen in Figure 1c. Here, there is a connection between regions B and C and an equivalence statement, B = C, is carried along. At the end of the sixth row, there is another connection between C and D (C = D) and all the regions are completed as seen in Figure 1d.

The system block diagram is shown in Figure 2. The pixels are read from the top of the image to the bottom from left to right. The delay line is represented by twenty (20) SI/SO CCD delay lines which are coded to obtain 100 colors (analog voltages) and obviate transfer efficiency problems. There are 20 levels of color comparisons for horizontal and vertical connections in the Coloring Operator. The Equivalence Box notes horizontal and vertical connections between different colors, recolors a pixel if necessary, and notes when a color is no longer being used thus activating the equivalence statement between two different colors. The column clock is actually fed to all the Feature Extractors and they indicate when a color is no longer used. The device which selects the appropriate Feature Extractor is a decoder. The Feature Extractors which accumulate the object features such as area and perimeter as well as the Scratch Feature Extractor form the basis for classification decisions.

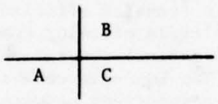
The desired data rate is 1 megapixel / sec. and to achieve this with surface channel CCD's with an assumed rate of 50 - 100 KHz requires Multiple Connected Component Operators in the same manner as multiple Median Filter, Gradient, or Non-Maximum Suppression Operators. That is, the line delay would be divided into a number of vertical columns. The object region must still be constructed across columns since an object region may be 100 pixels or 4 columns wide. One can begin to envision a hierarchy or branching structure of Connected Component Operators. The point here is that with the primitive operators, we were able to break the image into columns because the operators were relatively independent between columns. This is not the case here and it appears that one Connected Component Operator is desirable.

A ramification of one Connected Component Operator is that the pixels are moving through the delay line at the rate of 1 megapixel/sec. To insure numerical integrity, we shall organize the decoder taking the delay line output in the form of a Field Programmable Logic Array. The color (analog voltage) is quantized to ten levels and the highest bin containing a quantum of charge is identified and shifted into a delay line corresponding only to that bin. This means that 20 delay lines and a decoder are necessary to carry 100 colors.

We assume that the Coloring Operator processes a binary image, i.e. each pixel contains either a one (1) or a zero (0). The binary data stream will enter the Coloring Operator and emerge transformed into different colors or signal levels for different shapes. Since the image data is read out serially, the Coloring Operator is a local operator. The Coloring Operator is a transformation from a binary picture to a color one by a mapping T

$$T(A,B,M):C \rightarrow C^1$$

where C^1 is the color of the transformed pixel C, the variables A and B represent nearest neighbors of C, and M represents an available color the relative locations of pixels A, B, and C in the image plane area shown below.



We define the coloring window as always containing these three elements. Elements A and B are nearest neighbors of C and have already been processed by the Operator. Element B is located one horizontal line above elements C and A. Element C is painted by the Coloring Operator according to the following rule

For $C \neq 0$

$$C^1 = \begin{cases} A \text{ if } A \neq 0 \ B \neq 0 \\ B \text{ if } A = 0, B \neq 0 \\ M \text{ if } A = 0, B = 0. \end{cases}$$

When adjacent elements have different colors, the element being painted assumes the color of the nearest neighbor in the same line (rows dominate). Whenever elements A and B are zero and element C is not zero, element C^1 is given a new color.

To include multicolor capability, we want to access the Equivalence Operator whenever

For $C \neq 0$, $\quad A \neq B \neq 0$.

Now, we face problems such as which direction to actuate the equivalence statement, when to actuate it, and how to facilitate it.

The equivalence statement is actuated when one of the component colors is closed out;in order to continue a color (and not close it out), there must be a vertical connection somewhere along the next image line. One possibility is to construct a timer for each color such that a vertical connection would reset the timer. If the timer were allowed to reach 600 (image is 600 pixels long), it would enable the Equivalence Operator and close out the color. The timer could take the form of one of the channels of the large holding well found in the Feature Extractor for each color. At the rate that the pixels are shifted in the system, a quantum of charge would be entered in the timer channel; the accumulated charge would be non-destructively read out and compared to say 600. If the accumulated charge were larger than the amount equivalent to 600 shifts, the color would be closed out. If there were a vertical connection, the contents of the timer channel would be reset (if <600) to zero, and the accumulation started again for that color. Note that each color has its own timer and each is updated at every clock pulse. That component color which is closed out is directed into the color component of the equivalent statement which is not closed out.

There is one other logic step in implementing the equivalence statement and that is recoloring the previous line when a vertical connection between different colors is detected. This is necessary to form a connection between a tri-colored object region, for example. If we have the equivalence statements:

1 → 3
2 → 1.

Such an object will be treated as two object regions. Recoloring the previous line will restate the equivalent statements as:

3 → 1
2 → 1.

There is a Feature Extractor corresponding to each color; the signals from the Equivalence Operator enable the particular Feature Extractor for the equivalence computations. They also direct which Feature Extractor will receive the contents of the Scratch Feature Extractor, i.e. the color still active. The Feature Extractors themselves are visualized as a many-channeled, large holding well which follows along the line of this histogram-sorter. Each channel would correspond to a particular feature and since the features are linear, they would simply add in the Scratch Feature Extractor.

ACKNOWLEDGEMENT

The contributions of John Dehne and George Jones of NVL, Azriel Rosenfeld and David Milgram of Maryland, and Glenn Tisdale of Westinghouse to the implementation of this algorithm are noted here.

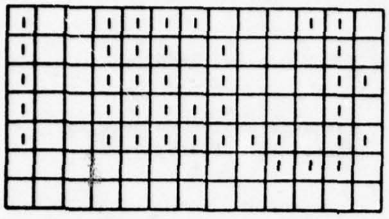


Figure 1 a. Binary Image

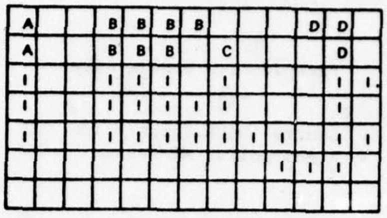


Figure 1 b. Computations for Second Row

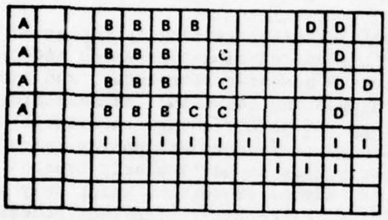


Figure 1 c. Computations for Fourth Row

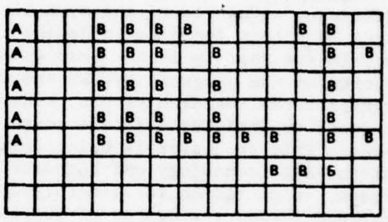


Figure 1 .d. Completed Image

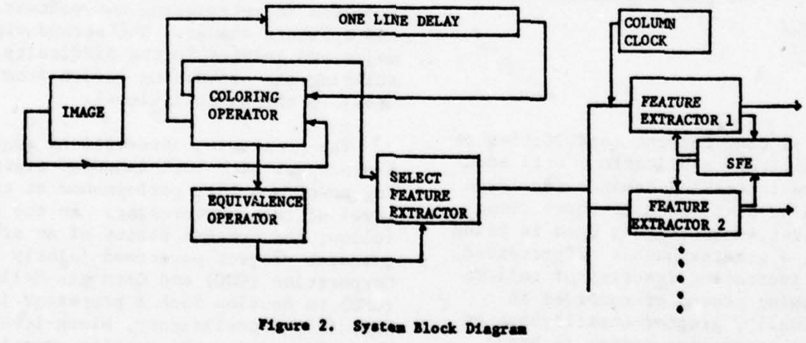


Figure 2. System Block Diagram

AN IMAGE PROCESSOR ARCHITECTURE

Pete Juetten & Gale Allen

Control Data Corporation
Minneapolis, Minnesota

Bob Hon & Raj Reddy

Carnegie-Mellon University
Pittsburgh, Pennsylvania

SUMMARY

Demands on image processing systems have been increasing as digital systems have been developed to support efficient, signal-level and symbolic-level processing, and this increase in demand is expected to accelerate sharply in the near future. This paper reports on the progress by CMU and CDC on a joint architectural research effort to develop a processor concept to anticipate future image processing needs. This processor is being designed, using state-of-the-art technology and automated design techniques, as an add-on for general-purpose host computers, and includes a number of functional units (with I/O and memory considered as functional units). The program-configurable structure is instrumental in providing both high parallelism and pipelining for throughput optimization. In order to provide the highest degree of system integrity for efficient computation, hardware and software are being developed concurrently. This includes a machine organization, a two-pass assembler, and a simulator.

INTRODUCTION

It is anticipated that current capabilities to process images for military applications will soon be inadequate. These increasing demands for image processing power can be attributed to three components of change. First, more imagery data is being collected to produce a greater number of processed images. Second, an increasing fraction of collected imagery data is being sensed or recorded in digital form, and finally, greater intelligence on the part of the image processing system is being demanded. These changes will require improvements of image processing systems in both signal-level processing and symbolic-level processing.

Signal-level processing is largely a matter of arithmetic manipulations, comprised of tasks such as transformation, interpolation , and filtering. These signal-level tasks are readily supported by currently available numeric, signal processors. To some degree, increasing signal-level demands can be met by brute-force increases in computer power with networks of available processors. Unfortunately, increases in the numbers of digital imagery sensors and image users tend to suggest that processing demands should increase by two orders of magnitude over the next few years. Meeting such an increase will have to be met by both an increase in signal-level performance and a shift of some of the burden from the signal-level to the symbolic-level.

Symbolic-level processing is concerned with the detection of and relationships among entities having certain semantic attributes, and involves symbol manipulating tasks such as searching, decision making, and path finding. Available numerically oriented processors are poorly suited for these tasks. This is particularly true of high performance, signal processors which rely heavily on pipelining, since their pipelines generally have to be flushed and refilled with each decision branch. While many non-numeric processors have been proposed, and a few built, there are two important disadvantages to attempting to satisfy increasing image processing demands through a network of numeric and non-numeric processors. First, the net would be inhomogeneous, leading to some problems in hardware maintenance and probably severe problems in interfacing and software development and software change. The second disadvantage is major and relates to the difficulty of smoothly shifting the processing burden from the signal-level to the symbolic-level.

The preceeding observations suggest a need for a processor with both improved signal-level computing power and high performance at the symbolic-level of image processing. In the paragraphs which follow, the current status of an architectural research project performed jointly by Control Data Corporation (CDC) and Carnegie-Mellon University (CMU) to develop such a processor is described. Thus far, a preliminary, block-level design has been adopted for the machine organization. In addition, software tools are being developed to support the design work and applications developments. Initial versions of an assembler and a register-level simulator have been constructed. These tools will be used to evaluate the performance of alternative hardware configurations.

GENERAL CONSIDERATIONS

In order to focus the architectural research effort on functional capabilities for a next generation image processor, the assumed application of the processor has been as an add-on to a general-purpose host computer, as illustrated by Figure 1. This environmental restriction not only allows a

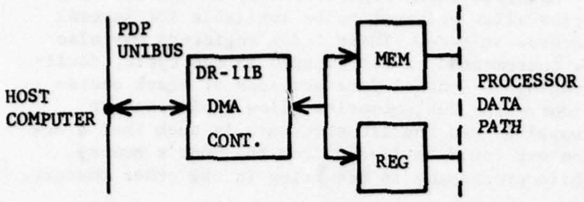


Figure 1. Connection to Host Computer

more productive hardware design effort, but substantially simplifies the simulator and software development. Little generality is lost through the restriction since augmenting the Input/Output performance for network applications will require modification of only two of ten substructures of the processor. The I/O capability has been limited to supporting buffered block transfers between host and processor memories, and no interrupt facilities have been provided. Other constraints which have been adopted in keeping with the goal of high signal-level and symbolic-level processing performance include a commitment to subnanosecond, ECL LSI technology and 16-bit integer words, with multiple precision and byte oriented capabilities. (No floating-point hardware is included.)

The main components of the processor at the block level are shown in Figure 2, and include 1) a set of function units, and 2) instruction memory and associated control logic. The processor operates in a sychronous manner with a basic cycle time of 15-20 nanoseconds. Instructions are fetched from the instruction memory at this rate and functional unit execution proceeds at the same rate. In most cases, two cycles are required to complete an instruction; during the first cycle the operands are fetched, and execution of the function occurs during the second cycle.

FUNCTIONAL UNITS

A collection of relatively autonomous functional units implement the arithmetic and logical functions of the machine. Each unit may execute a simple operation such as an add or a more complex operation which may take many cycles to complete. Most of the functional units have two inputs and one output, as well as inputs for the control of internal logic. The input operands are latched in local registers; this allows temporary results to be held as input operands for the next function without having to store them in data memory. Data memory is, in fact, controlled like any other functional unit.

Functional units included in the machine are: two adders, a multiplier, a logical unit, a barrel shifter/mask unit, an I/O box, and the control unit. In addition to appropriate operand registers, each functional unit has hardware to compare the result of an operation with a previously defined quantity held in an auxiliary register in the unit. The results of these comparisons are available at all times and may be sensed by the control unit for use in conditional execution of instructions.

CONTROL

The instruction memory is loaded by DMA transfer from the host computer. The instructions are subdivided into several fields: four fields for controlling four different functional units, a constant field which doubles as the destination address for a branch, and a field which controls the processor's data paths. Any instruction may initiate activity in four functional units simultaneously, allowing operand fetches to overlap the execution of functions.

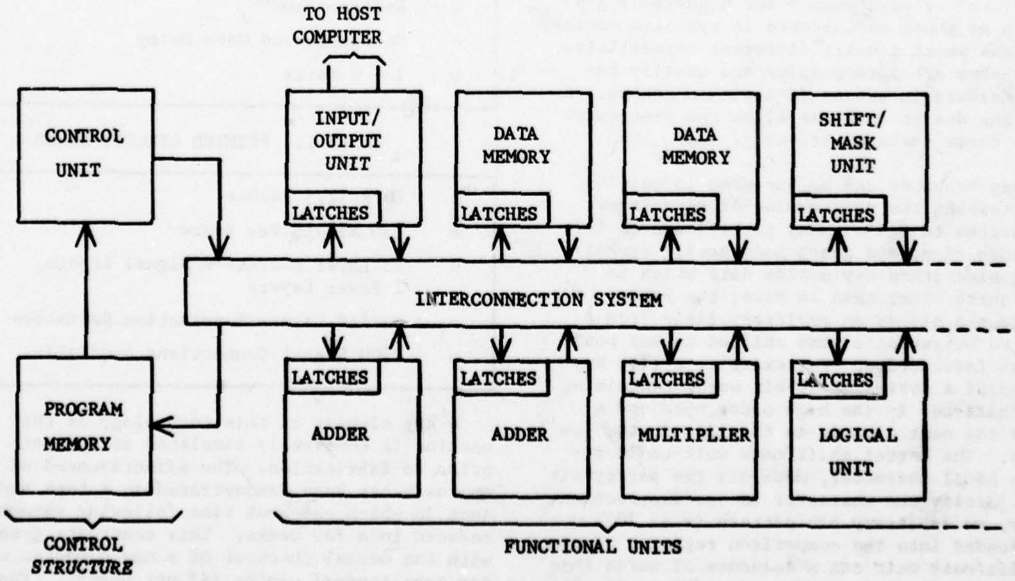


Figure 2. Preliminary Block Diagram

The execution of instructions is normally sequential. This may be altered by executing an appropriate function in a special functional unit called the control unit. The control unit is addressed the same as the other functional units, thus a branch instruction is performed by placing the proper code in one of the functional unit fields of the instruction. This unit also takes care of special instructions (e.g. HALT).

Instruction memory and data memory are both loaded via DMA transfer controlled by the I/O functional unit. The control registers in the I/O box may be loaded by either the processor or the host computer. This allows the host to load a program into the processor and start it. This also allows a program executing in the processor to overlay itself.

SIGNAL-LEVEL PROCESSING FACILITIES

Low-level image processing algorithms frequently contain short inner loops which must be executed for each pixel. In these cases, memory bandwidth limitations may dominate the total execution time for the algorithm. This problem is avoided by the fast data memories incorporated in the processor design. The presence of two independent memories allows the processor to acquire two new operands on each cycle, or to store a result and acquire an operand. The host machine may also be loading or retrieving data from these memories as the processor is running. Since an intermediate result can be held in the input register of a functional unit for the next operation, the number of memory accesses is reduced. By chaining operations in this fashion, it should be possible to execute many instructions between memory references.

SYMBOLIC-LEVEL PROCESSING FACILITIES

The processor design is intended to provide an effective level of performance for high-level processes such as those encountered in symbolic manipulation tasks which require different capabilities. Here algorithms are more complex and usually require a considerable amount of decision-making. A number of the design features allow the processor to execute these tasks efficiently.

Multiway branches can be executed in one instruction, easing the programming of algorithms having branches to one of many paths (such as for speech recognition, and graph searches). Symbolic processing algorithms may employ data which is packed in units other than 16 bits; the barrel shift/mask unit allows an arbitrary field from a data word to be extracted and shifted to any position in one instruction. For example, a list may be composed of a series of 16-bit words containing an ASCII character in the high order byte and a pointer to the next element in the list in the low order byte. The barrel shift/mask unit could retrieve the ASCII character, mask off the parity bit and right justify the character in one instruction. Similarily, an arbitrary bit pattern (e.g. 101) could be loaded into the comparison register of the barrel shift/mask unit and a sequence of words then searched for that pattern.

Multiple index registers (32) in the data memories allow pointers to be available for several records at once. These index registers may also be incremented or decremented in one cycle, facilitating sequential data accesses or stack operations. The dual memories allow rapid context swapping, and the I/O structure is such that a new context could be loaded from the host's memory while processing is occurring in the other context.

LOGIC COMPONENT TECHNOLOGY

The logic component technology for this processor, selected on the basis of performance and cost-effectiveness, is an emitter-coupled circuit developed by Control Data in conjunction with chip fabrication performed by Motorola and Fairchild. The integrated circuit chip characteristics are listed in Table 1. The fabrication process provides a semi-custom revision of LSI. With this process the cost of fabricating new chip types is much lower than with the custom LSI. In this technology the diffusion pattern is fixed and is the same for each chip type. A two-layer metalization interconnect is used to provide the variable structure. Features of the printed circuit board construction are listed in Table 2.

TABLE 1. ECL LSI ARRAY DESCRIPTION

- 168 ECL Gates Per Array
- 2 Input And-Nand Gates
- 165 x 175 Mil Die
- 4 Gates Per Cell
- 48 Signal Pins
- External Gates
- 6 Loads Per Output
- Collector Dotting
- Emitter "And"
- Subnanosecond Gate Delay
- 1 - 5 Watts

TABLE 2. PRINTED CIRCUIT BOARD

- 10 x 12.5 Inches
- 150 Arrays Per Board
- 15 Layer Boards--6 Signal Layers, 2 Power Layers
- Buried Layer--Termination Resistors
- 2100 Signal Connections Available

A key element of this technology is that the machine is completely simulated at the gate level prior to fabrication. The effectiveness of this approach has been demonstrated in a test bed project in which checkout time following assembly was reduced to a few weeks. This contrasts greatly with the normal checkout of a new computer which can take several months (if not years). Features of the simulation tools are listed in Table 3.

TABLE 3. AUTOMATED DESIGN TECHNIQUES

- o Logic Simulation
  - - 75,000 - 100,000 Gates
  - - 24 Array Types
  - - 10 Boards
  - - Gate, Foil, Coax Delays to 10 PS
  - - Worst Case Variation
- o Board Router
  - - Cyclical
  - - Interactive
  - - Photoplot Tape
  - - Simulator Input
- o Array Generator
  - - Digitized Array Layout
  - - Spacing Checks
  - - Mask Generation Tape
  - - Simulator Input

ASSEMBLER

The assembler runs on a CDC 6400 computer in batch mode, accepting standard 80-column punched cards as input. It produces an object code file as well as a 132-column line printer assembly listing. The assembly listing contains informative statistics, a listing of the cards input, object code produced, and any error messages. In order to facilitate rapid debugging, the syntax analyzer and code generator tag the offending token when an error is detected. Key features of the assembler are listed in Table 4.

The assembler translates a set of symbolic instructions into the appropriate machine instructions. A program in the assembly language is a number (1 or more) of instructions and assembler directives terminated by an "END" directive. Each instruction is composed of an optional label followed by one to four statements and terminated by a semicolon. A statement defines an operation for a specific functional unit and generally takes the form of an op code mnemonic followed by a number (possibly zero) of operands separated by commas. Each statement corresponds to one of the 15-bit functional unit fields in the machine instruction.

TABLE 4. ASSEMBLER FEATURES

- o Written In Fortran For Portability
- o Modularly Written For Ease Of Modification As The Processor Design Changes
- o Instruction Set Changes Implemented By Changing Only One Function Within The Assembler
- o Language Format (Syntax) Changes Implemented By Changing A Set Of Productions And Recompiling The Assembler
- o Machine Configuration Easily Changed
- o Input Text To The Assembler Is Completely Free Field: Banks, Comments, And Blank Lines May Be Mixed With Code To Improve Readability
- o One To Four Op Codes Allowed Per Instruction
- o Interconnect Routing Set Up Automatically By Assembler
- o Multiway Branch Blocks Automatically Moved To Power Of Two Boundaries
- o Errors Individually Flagged And Message Printed
- o Complete Assembly Listing Printed At Completion Of Assembly, Including A Symbol Table Dump And All Code Generated
- o Several Informative Statistics Are Maintained During Program Assembly And Printed At The End Of The Assembly Listing

Input text is free-format; instructions may cover several lines and spaces and tabs may be used freely to improve the readability of the code. The syntactic form of the processor programming language has been developed using a metalanguage with the following symbols. A class of objects is denoted by a word (usually descriptive) enclosed by angle brackets ("< " "> "). When such a class (e.g. <label>) is used, any one of the members of that class may be substituted for the occurrence of that class. Classes enclosed in square brackets ("[" "]") are optional - a member of the class may be present at that position but its presence is not mandatory. Using this notation an instruction could be represented as:

[<label>]<statement>[<statement>]
[<statement>][<statement>];

where <label> is any non-reserved word followed by a colon.

The assembler generates code in two passes. On the first pass (through the input deck) syntax errors are detected, statement fields are specified, some constant fields are assigned, labels are defined and a data path requirement is generated for each complete instruction.

The assembler makes the second pass through the code generated on the first pass after an END or ENDC is encountered. Constants which were unspecified after the first pass (e.g. forward branches) are assigned and the data path control field bits are set. At the end of the second pass the code generated is punched as an output deck. If an ENDC was the last directive, the symbol table and code generator are initialized and the assembler continues to read input, otherwise execution terminates.

The assembly example in Table 5 (follows text) is intended to provide some insight into the type of programming likely to be encountered with the processor. The program produces a summation of the integers from one to ten.

The code uses both adders to sum the integers from one to ten. Adder 1 is used to provide each input integer to adder 2 which calculates the running sum. I0 clears both operand registers of both adders. I1 latches a 1 in the B operand input register of adder 1; this 1 is used to increment the current integer (located in the A operand register) to find the next integer. I0 and I1 could not be combined into "ADD1 0,1 ,ADD2 0,0;" since this would require two different constants. I2 loads the terminating condition into the compare register of adder 1. During I3 the new integer (output of adder 1) and the last sum (output of adder 2) are added in adder 2. Simultaneously the integer is incremented in adder 1. I4 tests the terminating condition; adder 1 contains the <u>next</u> integer to be added, hence the test against 11 rather than 10. I5 is always executed because instruction fetches overlap instruction execution. At the end of I5 either I3 or I6 is executed, depending on whether the tenth integer has been added.

SIMULATOR

The CDC/CMU processor simulator is a software package designed to execute object code intended for the processor. The register level operations and results are identical to the hardware version. During the execution of the program a number of informative statistics are gathered. Table 6 lists major features of the simulator.

TABLE 6. SIMULATOR FEATURES

- Written In Fortran For Portability
- Modularly Written For Ease Of Modification
- Register Level Operations Identical To Those Of The Processor
- Load File Format Identical To That Of The Processor
- Run Time Statistics Maintained Include The Percent Utilization Of Each Functional Unit, Number Of Branches Executed, And Total Simulation Time
- Simulator Easily Reconfigured To Reflect Hardware Modifications
- Functional Unit Modules Follow A Standard Format, Allowing New Functions To Be Added To The Simulator With A Minimum Of Effort
- Break Point Feature Allows The State Of The Machine To Be Printed At User Selected Points During Program Execution

Each software functional unit is a separate module, containing a "start operation" function, a "conclude operation" function, and a function for accessing the condition codes. Internally the units are similar. The op code passed to the start function is checked for clock operand bits and the requisite operands are saved. Calling the finish function causes the result of the operation to be presented at the appropriate input and the condition codes to be set.

The simulator produces a listing which includes break-point printout and a statistics summary. Simulation of a program to sum the first ten integers resulted in the simulation listing of Table 7 (follows text).

ACKNOWLEDGEMENT

Several persons have contributed to this project in addition to the authors: Terry Hsu, Richard Kirchner, Joe Clarke, Walling Cyre, and Ken Enstrom from CDC; and Steve Rubin and Steve Saunders from CMU.

In addition, the project team is indebted to the many contributions of members of both staffs and in particular to those persons who contributed to HARP at CMU and the Flexible Processor at CDC.

TABLE 5. ASSEMBLY EXAMPLE

```
BEGIN PARSE AT 17.25.09.

        v  PROGRAM TO ADD THE INTEGERS FROM 1 to 10

                START 0                             v BEGIN EXECUTION AT LOCATION 0
        I0:     ADD1 0,0      ADD2 0,0;             v CLEAR BOTH ADDERS
        I1:     ADD1 ,1;                            v LOAD A 1 FOR INCREMENT
        I2:     LCMP ADDER1,13;                     v STOP WHEN WE REACH 10
        I3:     ADD2 ADDER1,ADDER2                  v ACCUMULATE THE SUM
                ADD1 ADDER1,;                       v INCREMENT
        I4:     BRNE ADDER1,I3;                     v BRANCH BACK IF NOT DONE
        I5:     NOOP;                               v ALWAYS EXECUTED
        I6:     HALT;                               v ADDER2 CONTAINS THE TOTAL

                ENDC

    END OF PARSE

SYMBOL TABLE:

            SYMBOL          ADDRESS

            I0              0000
            I1              0001
            I2              0002
            I3              0003
            I4              0004
            I5              0005
            I6              0006

CODE GENERATED:

   PC          FU0     FU1     FU2     FU3     CCNST      INTERCONNECTS REQUIRED

  0000        00061   00062   00000   00000   000000     77 17 17 17 17 77 77 77
  0001        00041   00000   00000   00000   000001     77 77 17 77 77 77 77 77
  0002        00101   00000   00000   00000   000013     77 17 77 77 77 77 77 77
  0003        00062   00021   00000   00000   777776     77 01 77 01 02 77 77 77
  0004        04100   00000   00000   00000   000003     77 77 77 17 77 77 77 77
  0005        00440   00000   00000   00000   777776     77 77 77 77 77 77 77 77
  0006        01040   00000   00000   00000   777776     77 77 77 77 77 77 77 77

START ADDRESS:  00000

INSTRUCTIONS GENERATED:   7     INST FIELDS USED:    9  of  28   CONSTANTS USED:  4 of 7

TOTAL ASSEMBLY TIME:   .22 SEC           ASSEMBLY COMPLETE AT 17.25.09.
```

TABLE 7. SIMULATION PROGRAM

```
BEGIN PARSE AT 08.40.45.

         v  PROGRAM TO ADD THE INTEGERS FROM 1 to 10

                  START 0
         I0:      ADD1 0,0        ADD2 0,0:       v CLEAR BOTH ADDERS
         I1:      ADD1 ,1;                        v LOAD A 1 FOR INCREMENT
         I2:      LCMP ADDER1,12;                 v STOP WHEN WE REACH 10
         I3:      BRNE ADDER1,I3;                 v BRANCH BACK IF NOT DONE
         I4:      ADD2 ADDER1,ADDER2              v ACCUMULATE SUM
                  ADD1 ADDER1,;                   v INCREMENT
         I5:      HALT:                           v ADDER2 CONTAINS TOTAL

                  ENDC

END OF PARSE
```

SESSION II

SYSTEMS

UNDERSTANDING NATURAL TEXTURE*

Joseph T. Maleson
Christopher M. Brown
Jerome A. Feldman

Computer Science Department
The University of Rochester

1. The Texture Problem

Texture is one of many cues available for identification of objects in scenes. Other cues are often sufficient for recognition: in a color image, the difference between sky and forest is most effectively made by describing principle hue. Sky is often "mostly blue and white" while forest is "mostly green and brown." Sometimes the distinction between areas is more subtle, such as the difference between a green lawn and green bushes. Here, the set of picture elements (pixels) that make up the lawn may be identical to the set of pixels that make up the bushes. The only difference between the two areas is the placement of the various pixels. The lawn and bushes present different two-dimensional patterns, and these kinds of patterns are what we mean by texture.

Computers deal with sampled images. A sampled image is a matrix of pixels. The data associated with a pixel may include intensity, hue, and saturation, as well as data outside of the visible spectrum such as infra-red or ultra-violet components, and information from other sources, like range data (describing distance from the observer). When data from more than one wavelength per pixel is available, the image is called multi-spectral.

Image segmentation is the division of an image into semantically meaningful regions, or segments. Image segmentation procedures have been most successful in multi-spectral images, where many cues are available. Most decisions can be made by looking at simple statistical measures of pixels in an area. Average intensity separates dark areas from light areas. Principal hue separates areas of different colors.

Because most objects in natural scenes differ from their neighbors in simple spectral measures, the use of texture as a cue has been neglected. All image segmentation projects include some kind of measure of textural properties and describe how they would make use of a better textural measure if they had one. A standard approach is to try to quantify coarseness. Edge per unit area is one such measure, used by [Ohlander, 1975] among others. This is an intuitively satisfying measure, and when used with other cues it has been shown to produce good results. Another approach has been to look at neighborhood adjacency matrices, and characterize textures by statistical properties of these matrices.

When the number of cues available in a scene is reduced, the importance of the textural cue increases. Aerial photographs and satellite images are prime examples of images where texture plays a key role in terrain classification. Often the textural difference is more subtle than a simple measure of coarseness. Other measures range from simple operations carried out in a small neighborhood, to Fourier techniques which transform the image into a frequency-space representation. None of these techniques provides a very satisfying (that is, successful) texture measure.

2. A Region-Based Texture Model

Any texture measure is inherently a statistical one. Statistics can be thought of as the science which allows optimal prediction of the true state of nature from several possible unknown states. In texture discrimination, one of two states of nature is true: either two textures are the same, or they are different. In texture classification, one of many possible textures is present at a particular area of an image, and the problem is to choose the appropriate one. Decision theory shows how to make the best choice, and determine a confidence level for that choice, when given appropriate statistical measures. Four textures difficult to discriminate are given in Figure 1.

The problem with statistical analysis is that if an inappropriate set of statistical measures is used, the final results are meaningless. For this reason, it is important to base statistics on a reasonable model of the phenomena to be measured. In the case of natural images, it is unreasonable to attempt to derive meaningful measurements from statistics based on features of individual pixels. The pixel is a unit intimately tied to the resolution at which an image has been scanned. The same texture scanned at two different resolutions (or equivalently, viewed from two different distances) should produce the same description, with the exception of factor of scale. Statistics over pixels, such as adjacency matrix calculations, actually give worse results at better (higher) resolution. At high resolution, a pixel is likely to be surrounded by pixels of similar value, except at edges, so that neighborhood statistics provide an expensive edge operator. The neighborhood measures were designed to provide information about spatial relationships

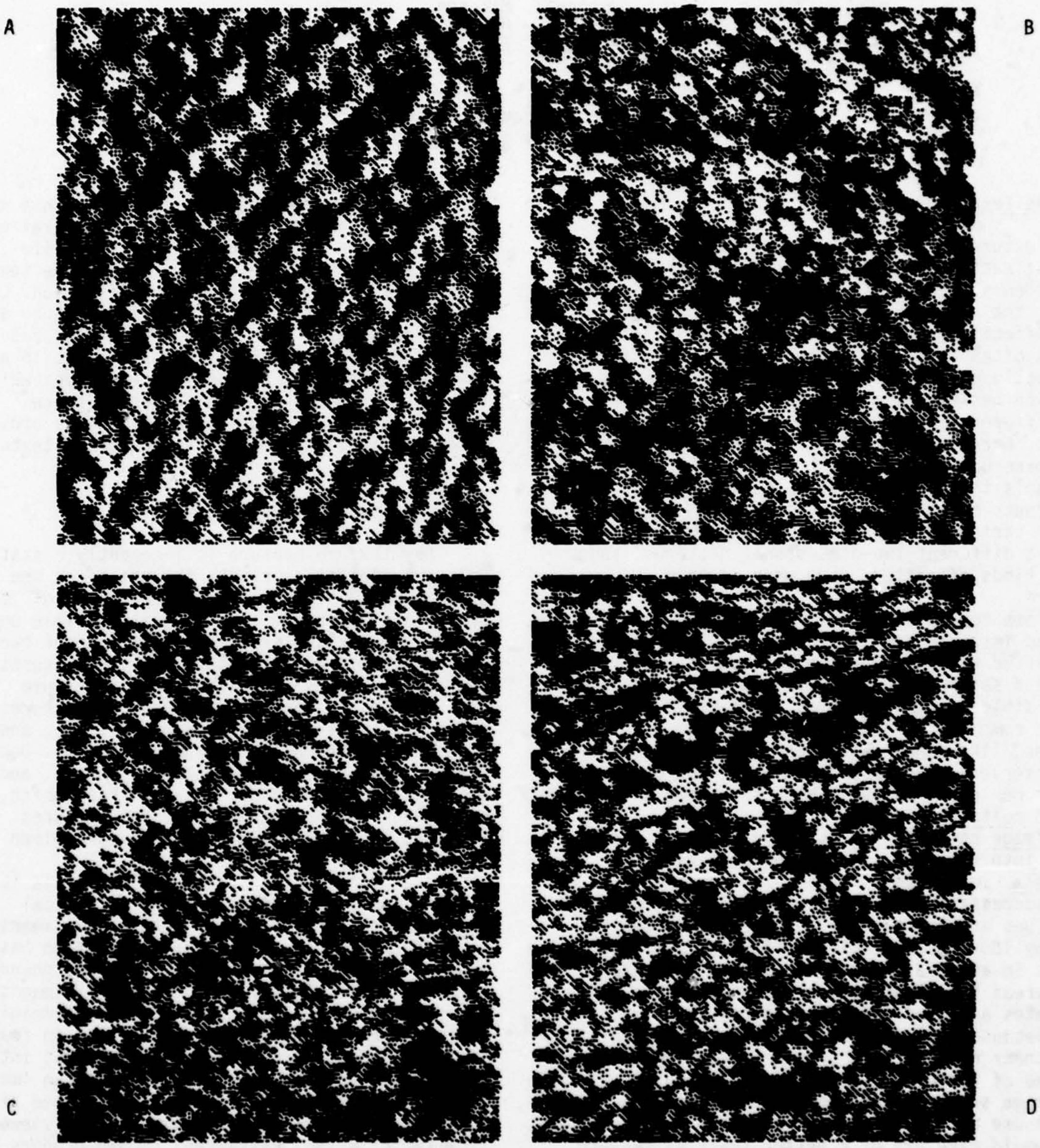


Figure 1: Four Textures Difficult to Discriminate:

A) Paper; B) Sand; C) Grass; D) Leather.

among pattern elements in textures. Julesz [1975] proposes that patterns made up of random dots can be described by enumerating the probability of finding an element of intensity "i" at distance "r" and angle "theta" from another element of intensity "j", for all i, j, r, theta. By limiting the allowable distance to 1 for computational efficiency, almost all spatial information is lost. In any case, Julesz was referring to patterns made up of uniform elements. In dealing with natural textures, the appropriate "pattern element" is not easy to define, and exact boundaries for elements are difficult to produce.

We have generalized the idea of spatial relationships in textured images by dealing with clusters of pixels, called "texture primitives." The texture primitives are restricted to clusters of "simple" shapes, which can be approximated by convex polygons. This restriction not only finesses the hard problem of complex shape description, but also provides a better basis for textural description. For example, a pattern made up of "T" shaped clusters is not greatly altered if the horizontal and vertical bars of the "T" are disconnected. This similarity is immediately revealed by representing the basic "T" element as two long elements with orthogonal axes, which are closest at the midpoint of one element and an endpoint of the other; the only difference lies in the exact distance between the elements. In natural textures, there is usually no unique pattern element, and important features, like orientation, can be successfully characterized by statistics of the individual texture primitives.

Our strategy, then, is to break an image down into texture primitives, examine local shape properties of these primitives, and examine a restricted set of spatial relationships among groups of these primitives. While our clustering strategy for producing texture primitives is limited here to a criterion of intensity closeness, the idea of the texture primitive is not limited to proximal intensity grouping. A texture whose basic elements are themselves textured can easily be included by using an appropriate texture measure to form texture primitives. This process does not lead us into any problems of infinite recursion because by the time a third level of indirection occurs (textured elements forming different textures which form elements for yet another texture), the patches which would have formed the basis for the highest-level texture may be considered objects in the image.

3. Local Properties of Regions

Information about textural orientation and scale could be captured using edge data. In general, edge analysis is a dual to region description: every region can be described by a set of edges, and every edge can be described as the boundary between two regions. The use of a regional representation is chosen because it is a more reasonable model of texture. Regions facilitate detection of classes of similar elements, as well as spatial relationships among elements.

Local analysis refers to measurements made on features of the texture primitives themselves. Textures made up of different primitive elements arranged in the same spatial order will appear different, with the degree of difference depending on the differences between primitives. Two kinds of features are important for classifying primitive elements. First is the similarity along the measure used to cluster the pixels; in the case of monochrome images, this is simply average intensity. Second is a description of shape features.

The important shape measures for the texture primitives are eccentricity and axial orientation. The eccentricity of a shape is a ratio of major to minor axis (computed as the principal axes of inertia). Every shape with non-zero eccentricity has its major axis as its orientation. Eccentricity and orientation are scale-invariant, and a measure of the distribution of region sizes is used to provide scaling information.

Many textures can be discriminated using only these kinds of local measurements. For example, a texture made up of long, thin regions at a particular orientation is immediately distinguished from textures with no prevalent orientation, or from textures with no long, thin regions. There is no combination of translation, rotation, or uniform scaling operations which can transform a highly-oriented texture to one which has no orientation preference. An important aspect of this approach to texture is that it provides a richer description than merely a point in a continuous n-dimensional feature space. Textures which cannot be rotated, translated, or scaled in 2-D to be similar can be classed as "absolutely different."

One simple discrimination strategy would be to separate textures into classes depending on some measure of average eccentricity, average region size, and axial orientation. For textures which are quite different, this kind of a technique is fast and efficient. For similar textures, however, this kind of strategy is prone to error. A much more powerful technique is available from local property data: examining the statistically dependent features, and characterizing the kinds of dependencies.

For regular patterns where particular elements occur frequently, a particular measure over all texture primitives in an image may not be significant, but that measure may be quite meaningful when used over a subset of the primitives. For example, a texture which contains long black rectangles on a noisy background may have low mean eccentricity, when averaged over all regions, but extremely high eccentricity in a small, low intensity range. Examining distributions of eccentricity, size, and axial orientation over different intensity ranges will produce a valuable description component when feature values are seen to cluster in different ranges. A texture whose large regions are mainly of low intensity will be different than one whose large regions are concentrated in the high intensity range (see Figure 2). In the images used here, intensity histograms have been normalized so that the number of pixels in each intensity range is the same. It is interesting to note that the apparent intensity range still differs, and appears to be based on the intensity of primitive elements which are important to the texture.

In naturally occurring textures, most of

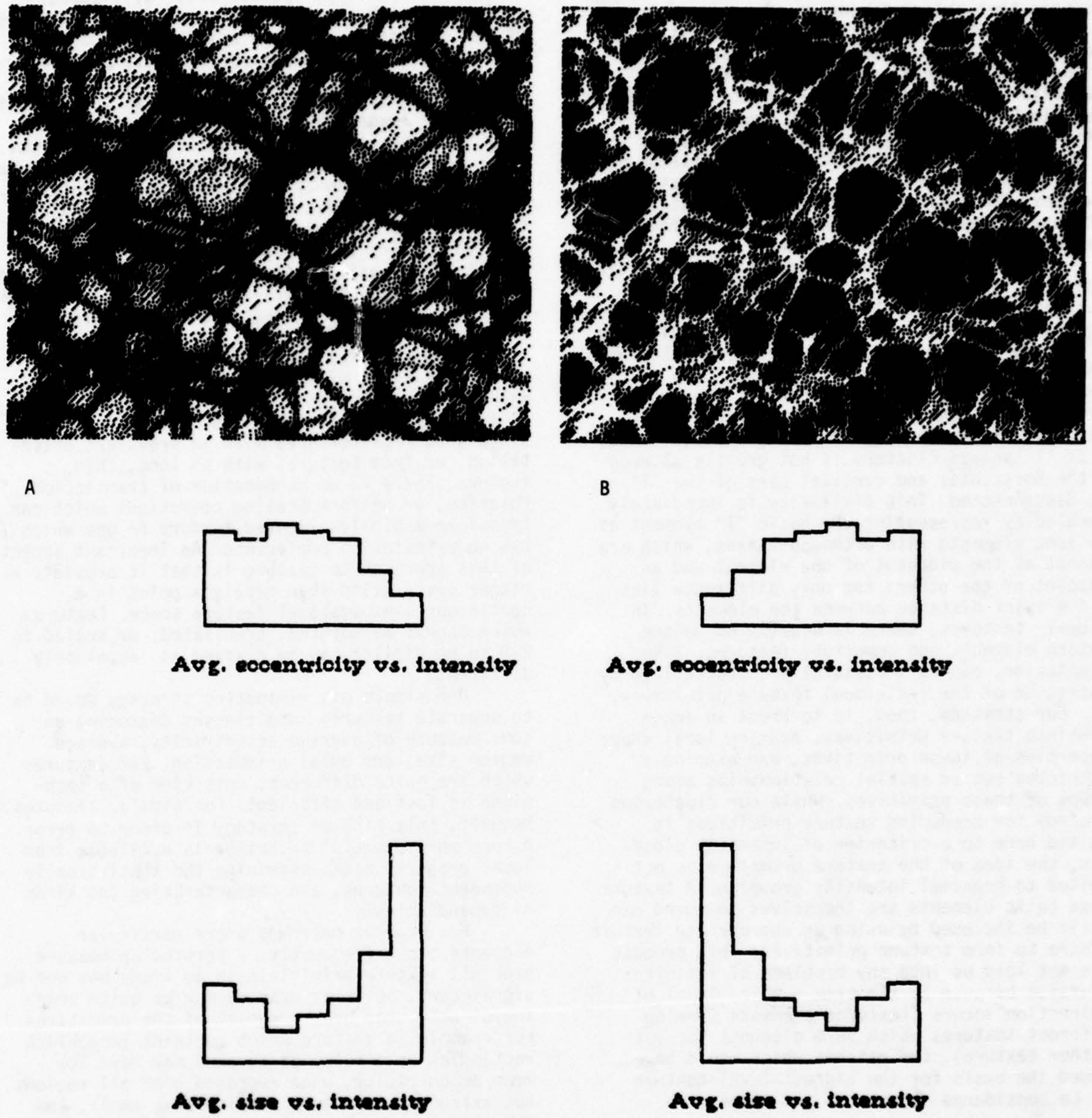


Figure 2: Identical Textures (A & B) with Intensity Reversal. Both have identical intensity histograms but average size or eccentricity vs. intensity range indicates differences.

the information available occurs in local measures. There is no particular spatial regularity of texture primitives in foliage, or gravel. In regular patterns, like a brick wall, the spatial relationships among regions will be necessary for discrimination among different regular kinds of bricklaying, but will not be very important to separate the brick from the bushes. In the more general problem of developing a theoretical basis for textures, it is important to deal with textures made up of the same primitives arranged differently. When local properties related to scale or orientation are removed, descriptions based only on local information will be inadequate.

4. Spatial Relationships Among Regions

One kind of relational measure is suggested by the Julesz spatial dependency matrix. While such a matrix may contain all of the important information for describing spatial relationships in a texture, it also contains much data that is not germane. Julesz's binary patterns have two kinds of primitive regions, but as the number of classes of primitive regions grows, the amount of data which is strictly an artifact of the underlying structure of a texture increases rapidly. For example, it is easy to describe a regular pattern of similar dots which are spaced regularly in both the horizontal and vertical directions. All that is necessary is the dot size and intensity, and the bi-directional periodicity. If a second grid of dots, with different intensity and period, is added to the first grid, a complex Moire pattern results. However, the underlying structure of the pattern is still easy to describe; it is simply the superposition of the two regular grids. Examining the complex interactions between the different kinds of dots produces a great deal of data, but no additional information. Spatial dependency matrices would not only include all of these interactions, but also many interactions between similar dots which are not related to the bi-directional regularity. What is needed is a characterization of only that spatial data which is meaningful to a texture.

The key to relational analysis is the construction of a useful set of relational features. The problem here is identical to all statistical problems: how to compress a great deal of data into a small amount of information. An enumeration of all spatial relationships extant in a texture is not useful: two areas of the same texture will seldom have exactly the same set of relationships, and it is not clear how to measure the distance between two sets of relationships. On the other hand, with statistical measures it is easy to distort the original data beyond recognition.

Usually, the kind of spatial relationships which allow human observers to discriminate among different textures can be described very simply. It is easy to discriminate between two dot patterns of equal average dot density, but where one texture requires that every pair of dots be separated by some minimum distance. The difference between these two kinds of dot patterns is certainly represented in the spatial dependency matrix, but use of the full matrix is data overkill. All of the data concerning angular separation between pairs of dots is extraneous. There is in fact only one number of interest, the radius of minimum separation.

We propose here that the interesting relationships are those between primitive elements in the same class, and that inter-class relations are usually artifacts. Several experimental paradigms demonstrate the tendency of humans to take cognizance of relationships among similar objects, and disregard the same statistical relationships among different objects. For instance, when a random dot pattern is turned a few degrees around its center, and printed on top of the unrotated pattern, circular formations are immediately recognized. If the second, rotated, pattern is presented in a different color, the circularity is not detected. Here, the spatial relationships among dots are not recognized unless the dots are all members of the same class.

The kinds of relational measures that we use yield data similar to some of the local operators. Colinearity is an important feature, and includes orientation data that is more global than the orientation measure provided by examining orientations of individual texture primitives. Here, for two eccentric regions to be colinear, their principal axes must be similar. Three noneccentric regions are needed before any measure of colinearity can be made. When eccentric regions are lined up along their minor axes, it seems more reasonable to describe the relationship as parallel. Other kinds of useful relationships which seem to be important include T-joints and V-joints.

The apparent nonlocal nature of relational analysis presents several problems. Standard global analysis would choose some window of an image, and look for a description of the relationships which exist inside of that window. This kind of approach is computationally costly and prone to several pitfalls. Because of the necessarily large domain over which relations are being computed, parallel processing models do not offer substantial time savings. Because some arbitrary window must be chosen, windows which span more than one textural area will produce meaningless results. Choice of an appropriate window size presents a dilemma.

Although relational measures describe statistics over a group of texture primitives, this does not mean that the benefits of local computation must be lost. Instead of looking for relationships within a window, we look for relationships for each primitive within a neighborhood whose shape is determined by the shape of the primitive. That is, a particular eccentric region can belong to at most one colinear set, and one parallel set. The likelihood of a particular relationship can be measured for every region in the neighborhood, and if the best likelihood is higher than some threshold, the existence of the relationship is posited. By choosing a small set of relations that are recognized, the relational data for a particular primitive element becomes simply another feature of that element. The same kind of techniques used to describe textures (partially) by using shape features of texture elements can be extended to the new relational features. Arbitrary, non-linearly separable textural boundaries can still be found by clustering texture elements into compatible sets.

An example of the use of relational measures is discrimination between water and straw, when the appropriate scaling and rotational normalizations have been executed. The resulting textures are quite similar, but the straw has many regions which are colinear along the major axis of orientation. The water contains regions which are parallel along an axis which is unrelated to the orientation of individual regions, but dependent on the illumination angle. Figure 3 shows these textures, and Figure 4 the processing steps.

These simple relations add enough power to the local properties already described to differentiate among most natural textures. When discrimination fails, the textures which have been unsuccessfully classified are practically indistinguishable to a human observer, and relate to irregularities within a single texture. If the textural classification of surrounding areas were used as an additional cue in hard cases, near perfect classification would be possible.

5. Utilizing Texture Measures: Design Decisions

Successful statistical analysis depends on a realistic model of the population from which data has been gathered. By choosing a reasonable, region-based model of texture, it is possible to provide descriptions which are useful in a wide range of image segmentation problems. The most important decision was to represent textures by analyzing what kind of clustering occurs among pixels, rather than trying to collect a set of features for each pixel.

In any problem domain it is important to remain flexible in spite of permutations in the given problem. A very different set of images might call for a slightly different set of features. In fact, given any strategy for discriminating texture, it will be possible to construct textures which cannot be discriminated using that strategy, but which will be discriminated using another approach. For example, textures made up of triangles will not be discriminable using the set of features described here from another texture in which the triangles have been replaced by equivalent area squares. It is easy to see what feature would be sufficient for discrimination in this case. One example of two textures not discriminable by humans is given by Julesz in his figure contrasting randomly distributed "R's" with randomly distributed mirror-image "R's." The immediate impression is that the two textures are identical, but an analytical examination will quickly find the boundaries between the two textures.

When constructing a set of textural descriptors, the nature of the problem domain must be considered. More information will only produce better results in any reasonable system, and the system described here is not an exception to that rule. At every stage of computation where it is possible to use knowledge already gained to predict the most effective subsequent computation, that knowledge should be used.

6. Applications and Related Work

Texture can be an important cue for segmentation in many problem domains. Terrain classification is often only possible with textural descriptors. Natural scenes are full of textured areas, and successful segmentation of natural images has been possible only by limiting the textural content of input scenes. Using texture description procedures will improve the performance of segmentation programs, yielding more accurate region assignments, and reducing the computing time required for textured areas.

Texture often provides additional cues, like surface orientation or depth information. These cues are called texture gradients. Texture gradient information can be extracted from textural description, using a realistic model of the transformations caused by altering orientation. Some kinds of transformations are due to three-dimensional phenomena, and require different models for what "texture gradient" means. Black dots painted on a white sphere will undergo a particular kind of shape transformation as they move away from the sphere center. Black spikes protruding from a white sphere will give the appearance of the same texture in the center of the sphere, but a very different textural transformation indicates distance from the center. In both cases, the texture gradient will be reflected in predictable changes of both local and relational statistics of texture primitives. One direction for further research is to set up appropriate models for such gradient transformations, and produce algorithms which detect these changes.

Depth cues from textural transforms present the problem of textural artifacts produced by low-resolution scanning. As a texture recedes, it will eventually reach the point that the scanning resolution is insufficient to detect important texture primitives. Intuitively, a texture should become less coarse as it recedes, since the primitive elements become smaller. But the sampling problems often cause fine textures to become coarser as they recede. Again, it is possible to predict the distance at which such anomalies occur, and use neighboring texture information to hypothesize a continuous depth change across an apparent textural break.

Texture is a high-resolution phenomenon. It would be useful to provide input sensors with the ability to monitor an area of textural interest at high resolution, and produce a textural description of that area. This would make textural cues as easy to use as multi-spectral information, and add considerable power without sacrificing the benefits of lower-resolution scanning for general segmentation. While low-resolution images may appear to contain textural areas, the range of such textures is necessarily small, and often reflects artifacts of the scanning resolution. The appearance of varied textures in low-resolution images is largely due to semantic prediction of what texture ought to be present in a recognized image. Recognition of low-resolution faces is one example of the brain's excellent capacity to produce effective hypotheses with little input information. Trying to find texture in low-resolution images is a little like trying

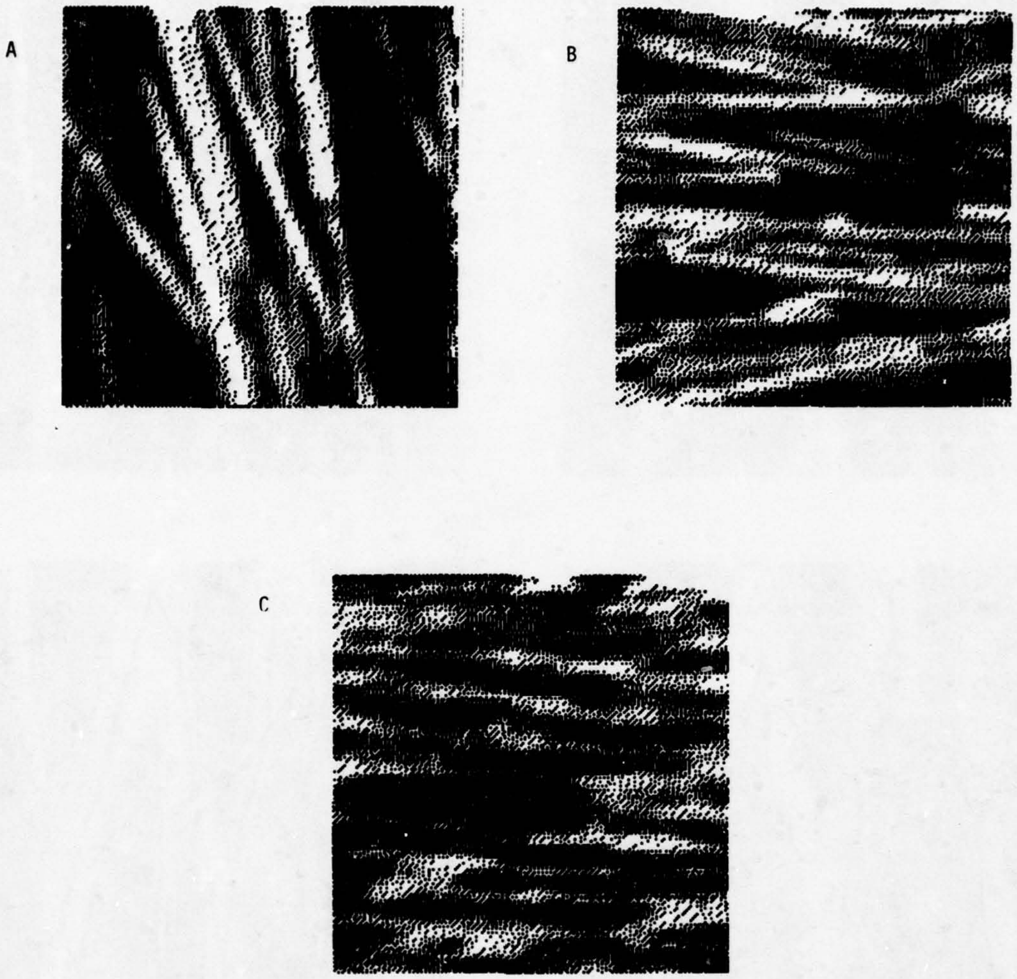


Figure 3: A straw texture (A) may be normalized (B) to appear similar to a water texture (C). Normalized straw and water may be discriminated by region colinearity.

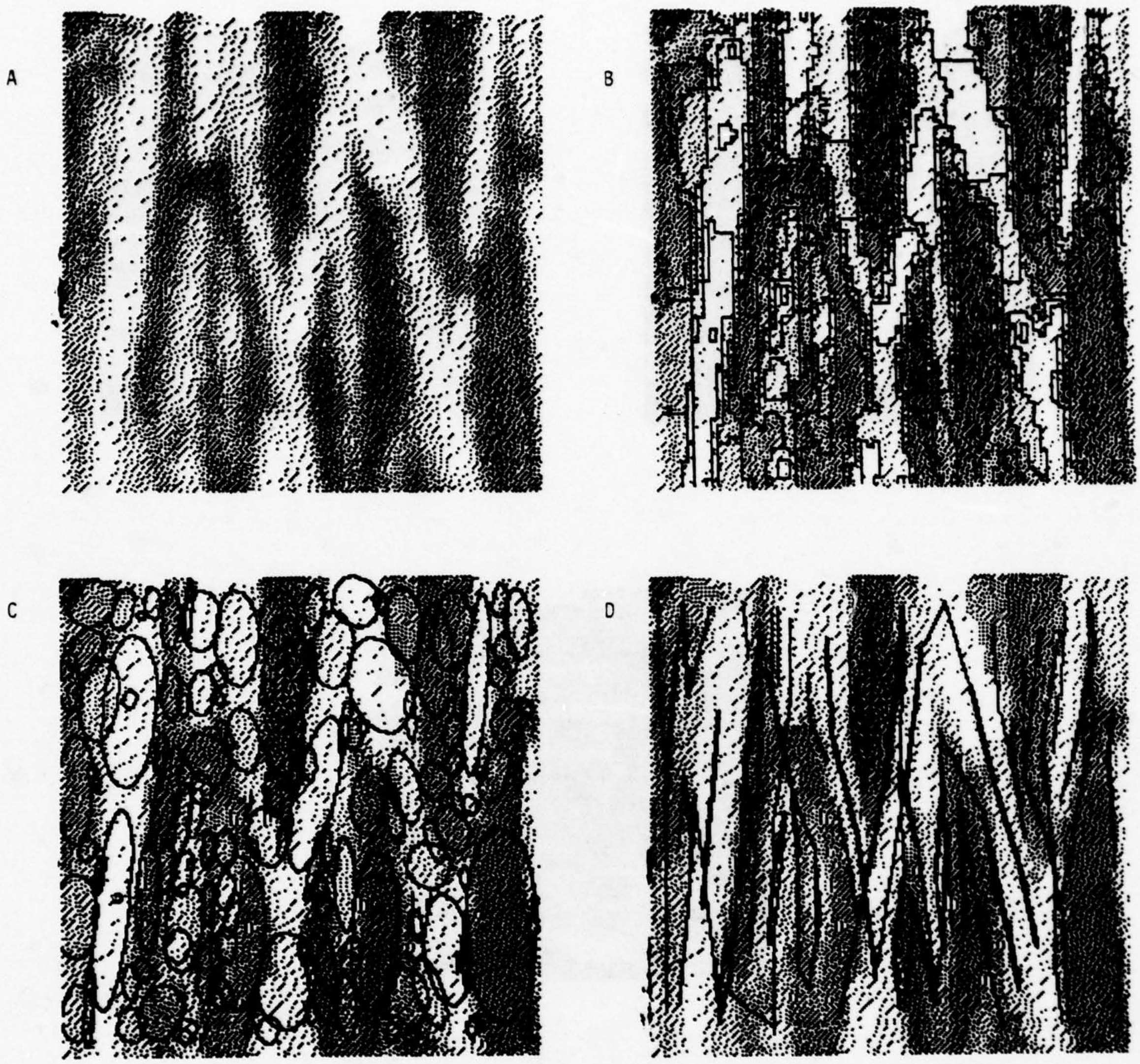


Figure 4: An 80x80 section of straw texture (A), its region boundaries (B), elliptical approximation to regions (C), and colinear regions connected center-to-center (D).

to predict color from a black and white image. Although a person might predict many colors quite successfully, there is no reason to believe that such color information is contained in the raw input data.

7. Summary

Texture analysis is a tool to detect the forest from the trees. Understanding texture does not solve the image understanding problem, nor does it produce the ultimate region grower. Texture descriptions provide a useful input to higher-level understanding programs, and a valuable cue for many segmentation problems. We have presented here a technique for producing useful descriptions of texture based on local shape statistics of simple regions called texture primitives, and on a simple set of relationships among similar classes of texture primitives. The usefulness of this description has been demonstrated through standard discrimination/classification tests. More importantly, these descriptions provide quantification of discrimination criteria and allow recognition of similar textures over transformations of scale, intensity, and rotation. Some different textures can be normalized to produce descriptions which are as similar as possible, and these normalized textures do in fact appear quite similar. Proposed extensions include using a world model to detect spatial orientation and distance transformations.

REFERENCES

Julesz, B., "Experiments in the Visual Perception of Texture," Scientific American 232, April, 1975.

Ohlander, R. "Analysis of Natural Scenes," Ph.D. Dissertation, Carnegie-Mellon University, Pittsburgh, 1975.

*This paper reports work done by Joseph T. Maleson; further details may be obtained in his forthcoming thesis.

RELAXATION METHODS: RECENT DEVELOPMENTS

Azriel Rosenfeld

Computer Science Center, University of Maryland, College Park, Maryland 20742

ABSTRACT

This note summarizes two recent studies on "relaxation" techniques for image segmentation:

1) The use of hierarchical discrete relaxation for waveform parsing

2) The estimation of coefficients for probabilistic relaxation processes by statistical analysis of input images.

This work was done by the Computer Vision Laboratory of the University of Maryland under the support of National Science Foundation Grant MCS-76-23763.

INTRODUCTION

"Relaxation" is an iterative approach to classifying a set of interrelated objects (e.g., parts of an image). In "discrete relaxation", a set of possible class names is initially associated with each object. At subsequent iterations, class names are discarded from an object if they are inconsistent with the surviving class name possibilities for other, related objects; this is done "in parallel", for all objects simultaneously. The process is repeated until no further changes can take place; it often yields highly unambiguous classifications. Waltz [1] applied this approach to labeling the parts of a line drawing; a general discussion of such methods can be found in [2].

In "probabilistic relaxation", a set of estimates of class assignment probabilities is initially associated with each object. At subsequent iterations, the probabilities are adjusted in accordance with the support that they receive from the class probabilities of related objects. This process, when repeated, often leads to a marked reduction in the ambiguity of the classifications. A general discussion of such methods can be found in [2, 3], and a collection of examples, involving the labeling of image points for segmentation or noise cleaning purposes, is reviewed in [4].

This note summarizes two recent studies on relaxation methods:

1) The use of hierarchical discrete relaxation for waveform parsing

2) The estimation of coefficients for probabilistic relaxation processes by statistical analysis of input images

HIERARCHICAL RELAXATION FOR WAVEFORM PARSING

Images and waveforms can often be described hierarchically as consisting of parts that are in turn composed of subparts, and so on, down to a level of "primitive" parts; where at each level, the parts are in approximately specified positions relative to one another. Such a hierarchical structure is essentially a layered system of "spring-loaded" templates [5]. The process of recognizing that a description of this type applies to a given image or waveform is essentially a process of parsing with respect to a stratified context-free grammar, with the primitive parts as terminal symbols.

In [6], a parallel, iterative method (called a "relaxation" method) of detecting spring-loaded template matches was proposed. In this method, matches to the subtemplates are detected, and for each such match, supporting evidence is sought -- i.e., do other matches occur in the expected relative positions. Subtemplate matches for which sufficient evidence is lacking are discarded, and the process is iterated (since discarding one subtemplate may weaken the evidence for another one). This process can be carried out in parallel for all the subtemplate matches, so as to rapidly eliminate all but those that belong to matches of the entire spring-loaded template.

This relaxation approach can be generalized to hierarchical spring-loaded templates. Here the interactions among the parts are more complicated; when a part is discarded at one level, this can cause other parts to be discarded at other levels, and further iteration may be needed at all of these levels. The procedure involves the following steps [7]:

1) Primitive detection, to create the lowest layer of the hierarchical network; or, more generally, given the nth layer, creation of the (n+1)st layer by detecting matches to the appropriate set of templates.

2) Elimination of nodes from any layer if they do not contribute to nodes in the following layer.

3) Elimination of nodes within any layer if they do not occur in the proper context, as defined by the templates that are applicable to that layer.

This procedure was applied to a noisy waveform defined by a four-layer template hierarchy. The primitive detection process found 25 possible primitives in this waveform, but at successive stages of the procedure, most of these were eliminated, until only those corresponding to the ideal waveform remained, and the correct "parse" of the waveform (in terms of the template hierarchy) was obtained. Extensions and further applications of this approach are planned, as discussed in [7].

COEFFICIENT SELECTION FOR PROBABILISTIC RELAXATION

In a probabilistic relaxation process, the class probabilities for each object are adjusted in accordance with the support that they receive from the class probabilities of other objects. This support is usually defined in terms of a set of coefficients, one for each pair of (object, class) pairs. A positive coefficient indicates that these pairs mutually reinforce, a negative coefficient indicates that they conflict, while a near-zero coefficient indicates a "don't-care" situation.

In earlier work [4], it is assumed that these coefficients can be defined from an understanding of the given classification problem. For example, consider the problem of detecting smooth curves in an image. Here each image point can belong to a set of classes corresponding to the existence of curves through the point at various orientations, or to no curve at the point. The curve probabilities in given orientations at two points should support one another if curves in those orientations through the two points would smoothly continue one another. Coefficients representing this support can be defined in various ways. For some choices of the coefficients, the process is highly successful in enhancing smooth curves while suppressing noise, while for other choices it is less successful.

In [8], a method of automatically defining coefficients for probabilistic relaxation processes is introduced, based on statistical analysis of the initial class probabilities. In particular, given a pair of (object, class) pairs (O_1,C_1) and (O_2,C_2), let E_1, E_2 be the events that O_1 is in C_1 and O_2 in C_2, respectively. The mutual information of this pair of events has just the properties that are desirable in relaxation coefficients: it is high if E_1 and E_2 tend to co-occur, and low if they do not. We can estimate the mutual information for each such pair of events, by statistically analyzing a suitable ensemble of input data, and use the estimated values as relaxation coefficients.

This approach was applied in [8] to the curve enhancement problem. For each point P of an input image, initial probabilities of there being curves in various orientations through P, or no curve at P, were derived by normalizing the outputs of line detection operators applied at P. Mutual information was then computed for all possible pairs of neighboring points and curve orientations (or no curve), averaged over all points of the image. When the resulting mutual information values were used as coefficients in a curve enhancement relaxation process, good results were obtained. The results remained good when coefficients derived from one image were applied to enhance curves in an image of an entirely different type. These experiments suggest that it should be possible, in many cases, to derive relaxation coefficients automatically by statistically analyzing a suitable ensemble of input data.

REFERENCES

1. D. Waltz, Understanding line drawings of scenes with shadows, in P. H. Winston, ed., The Psychology of Computer Vision, McGraw-Hill, New York, 1975, pp. 19-91.

2. A. Rosenfeld, R. Hummel, and S. W. Zucker, Scene labelling by relaxation operations, IEEE Trans. Systems, Man, Cybernetics SMC-6, 1976, 420-433.

3. J. M. Tenenbaum and H. G. Barrow, IGS: a paradigm for integrating image segmentation and interpretation, in C. H. Chen, ed., Pattern Recognition and Artificial Intelligence, Academic Press, New York, 1976, pp. 593-616; also in Proc. 3rd Intl. Joint Conf. on Pattern Recognition, Coronado, CA, Nov. 1976, 504-513.

4. A. Rosenfeld, Iterative methods in image analysis, Proc. IEEE Conf. on Pattern Recognition and Image Processing, Troy, NY, June 1977, 14-18.

5. M. A. Fischler and R. A. Elschlager, The representation and matching of

pictorial structures, IEEE Trans. Computers C-22, 1973, 67-92.

6. L. S. Davis and A. Rosenfeld, An application of relaxation labeling to spring-loaded template matching, Proc. 3rd Intl. Joint Conf. on Pattern Recognition, Coronado, CA, Nov. 1976, 591-597.

7. L. S. Davis and A. Rosenfeld, Hierarchical relaxation for waveform parsing, University of Maryland Computer Science Center Technical Report 568, August 1977.

8. S. Peleg and A. Rosenfeld, Determining compatibility coefficients for relaxation processes, University of Maryland Computer Science Center Technical Report 570, August 1977.

A STEREO VISION SYSTEM

Donald B. Gennery

Computer Science Department
Stanford University
Stanford, California 94305

Abstract

Several techniques for use in a stereo vision system are described. These include a stereo camera model solver, a high resolution stereo correlator for producing accurate matches with accuracy and confidence estimates, a search technique for using the correlator to produce a dense sampling of matched points for a pair of pictures, and a ground surface finder for distinguishing the ground from objects, in the resulting three-dimensional data. Possible ways of using these techniques in an autonomous vehicle designed to explore its environment are discussed. Examples are given showing the detection of objects from a stereo pair of pictures, including some examples using aerial photographs.

Introduction

This paper describes a stereo vision system for use by a computer-controlled vehicle which can move through a cluttered environment, avoid obstacles, navigate to desired locations, and build a description of its environment. One possible application of such a vehicle is in planetary exploration. Our experimental vehicle is described in [4].

As the vehicle moves about, it takes stereo picture pairs from various locations. This could be done with two cameras mounted on the vehicle, but with our present vehicle with one camera, it is done with the vehicle at two locations. Each of these stereo pairs is processed to extract the needed three-dimensional information, and then this information from different pairs can be combined in further processing.

The processing of the stereo pairs is done as follows. First, an interest operator finds small features with high information content in the first picture. Then, a binary search correlator finds the corresponding points in the other picture. (The interest operator and the binary search correlator were both developed by Moravec [4].) Next, a high-resolution correlator is given these matched pairs of points. It tries to improve the accuracy of the match, and it produces an accuracy estimate in the form of a two-by-two covariance matrix, and a probability estimate giving the goodness of the match. The coordinates of these matched points are corrected for camera distortion as described by Moravec [4]. A stereo camera model solver then uses these matched pairs of points to find the five angles that relate the position and orientation of the two camera locations. The accuracy estimates are used by the camera model solver to weight the individual points in the solution and to compute accuracy estimates of the resulting camera model. A dense sampling of points is now matched over the pictures. The known camera model is used to restrict the search for these matches to one dimension, and by first trying matches approximately the same as neighboring points that have already been matched, often no search is needed. In any case, the precise matches are produced by the high-resolution correlator, and its probability estimates are used in guiding the search. After these matched points are corrected for camera distortion, distances to the corresponding points in three-dimensional space are computed, using the known camera model. The accuracy estimates of the matches and of the camera model are propagated into accuracy estimates of the computed distances. The three-dimensional information for all of the matched points is now transformed into a coordinate system approximately aligned with the horizontal surface. (The high-resolution correlator, the stereo camera model solver, and the technique for producing the dense sampling of matches are described later in this paper.)

Information from more than one stereo pair can be combined to produce a more complete mapping of points over the area. A ground surface finder is then used to find the ground for portions of the scene, which may be tilted slightly relative to the assumed horizontal coordinate system. (The ground surface finder is described later in this paper.) Points which lie sufficiently above the ground surface can be assumed to lie on objects. (In the process of finding the ground surface and finding objects, the accuracy and probability estimates are useful.)

Stereo Camera Model Solver

If the image plane coordinates of several pairs of corresponding points in a stereo pair of images have been measured, it is possible in general to use this information to compute the relative position and orientation of the two cameras, except for a distance scale factor. Once this calibration has been performed, the distance to the object point represented by each pair of image points can be computed.

A procedure that performs the above stereo camera model calibration by means of a least-squares adjustment has been written. It includes automatic editing to remove wild points, the use of a two-by-two covariance matrix for each point for weighting purposes, estimation of an additional component of variance by examination of the residuals, and propagation of error estimates into the results.

Consider any point in the three-dimensional scene. Let the coordinates of the image of this point in the Camera 1 film plane be x_1,y_1 and the coordinates of its image in the Camera 2 film plane be x_2,y_2. Image point x_1,y_1 corresponds to a ray in space, which, when projected into the Camera 2 film plane, becomes a line segment. The distance (in the Camera 2 film plane) from image point x_2,y_2 to the nearest point in this line segment is the magnitude of the error in the matching of this point. This error is a function of the angles which define the relative position and orientation of the two cameras. (These angles are the azimuth and elevation of the position of Camera 2 relative to the position of Camera 1, and the pan, tilt, and roll of Camera 2 relative to the orientation of Camera 1.) The camera calibration is done by adjusting these angles to minimize the weighted sum of the squares of these errors for all of the points that are used. Since the problem is nonlinear, the procedure uses partial derivatives to approximate the problem by the general linear hypothesis model of statistics, and iterates to achieve the exact solution.

The automatic editing is done as follows. First, a weighted least-squares solution as described above is done using all of the points. Then the point which has the largest ratio of residual to standard deviation of the residual is found. This point is tentatively rejected, and the solution is recomputed without this point. If this point now disagrees with the new solution by more than three standard deviations, it is permanently rejected, and the entire process repeats. Otherwise, the point is reinstated, and the process terminates. However, if an F test comparing the computed and given values of the additional variance of observations shows the solution that includes the point to be bad, the point in question is rejected in any event.

A more complete description of the camera model solver can be found in [1].

High-Resolution Correlator

Consider the following problem. A pair of stereo pictures is available. For a given point in Picture 1, it is desired to find the corresponding point in Picture 2. It will be assumed here that a higher-level process has found a tentative approximate matching point in Picture 2, and that there is an area surrounding this point, called the search window, in which the correct matching point can be assumed to lie. A certain area surrounding the given point in Picture 1, called the match window, will be used to match against corresponding areas in Picture 2, with their centers displaced by various amounts within the search window in order to obtain the best match.

Thus when the matching process (correlator) is given a point in one picture of a stereo pair and an approximate matching point in the other picture, it produces an improved estimate of the matching point, suppressing the noise as much as possible based on the statistics of the noise. It also produces an estimate of the accuracy of the match in the form of the variances and covariance of the x and y coordinates of the matching point in the second picture, and an estimate of the probability that the match is consistent with the statistics of the noise in the pictures, rather than being an erroneous match. This probability will be useful in guiding a higher-level search needed to produce a dense sampling of matched points.

Let $A_1(x,y)$ represent the brightness values in Picture 1, $A_2(x,y)$ represent the brightness values in Picture 2, x_1,y_1 represent the point in Picture 1 that we desire to match, x_2,y_2 represent the center of the search window in Picture 2, w_m represent the width of the match window (assumed to be square), and w_s represent the width of the search window (assumed to be square), where x and y take on only integer values.

The following assumptions are made. A_1 and A_2 consist of the same true brightness values displaced by an unknown amount in x and y, with normally distributed random errors added. The errors are uncorrelated with each other, both within a picture (autocorrelation) and between pictures (cross correlation), and the errors are uncorrelated with the true brightness values. (The asuumptions concerning errors hold fairly accurately for the usual noise content of pictures. The assumption concerning the true brightness values will be relaxed slightly below to allow bightness bias and contrast changes. However, another type of change is perspectve distortion, which can be important with large match windows, but it will not be discussed here.)

We temporarily assume that the variance of the errors is known for every point in each picture.

We now wish to find the matching point x_m,y_m which will produce the best match of $A_2(x+x_m-x_1,y+y_m-y_1)$ to $A_1(x,y)$ in some sense. Traditionally the match which maximized the correlation coefficient between A_1 and A_2 has been used [2]. Indeed, this is a reasonable thing to do if one of two functions has no noise. However, here both functions have noise. This fact introduces fluctuations in the cross-correlation function which may cause its peak to differ from the expected value. *Ad hoc* smoothing techniques could be used to reduce this effect, but an optimum solution can be derived from the assumed statistics of the noise.

Let ϵ represent the w_m^2 - vector of the differences $A_2(x+x_m-x_1,y+y_m-y_1) - A_1(x,y)$ over the w_m by w_m match window, for a given trial value of x_m,y_m, and let x_c,y_c represent the true (unknown) value of x_m,y_m. Let P represent a probability and p represent a probability density with respect to the vector ϵ. Then by Bayes' theorem

$$P(x_m,y_m=x_c,y_c \mid \epsilon) = \frac{P(x_m,y_m=x_c,y_c)\, p(\epsilon \mid x_m,y_m=x_c,y_c)}{\Sigma\, P(x_m,y_m=x_c,y_c)\, p(\epsilon \mid x_m,y_m=x_c,y_c)}$$

If we assume that the *a priori* probability $P(x_m,y_m=x_c,y_c)$ is constant over the search window and is zero elsewhere, this reduces to

$$P(x_m,y_m=x_c,y_c \mid \epsilon) = k\, p(\epsilon \mid x_m,y_m=x_c,y_c)$$

where k is any constant of proportionality. Since ϵ consists of uncorrelated normally distributed random variables,

$$\begin{aligned} p(\epsilon \mid x_m,y_m=x_c,y_c) &= k\, \Pi \exp(-\frac{.5\, \epsilon_i^2}{\sigma_1^2+\sigma_2^2}) \\ &= k \exp(-\frac{.5\, \Sigma\, \epsilon_i^2}{\sigma_1^2+\sigma_2^2}) \\ &= k\, w \end{aligned}$$

where

$$w = \exp(-.5\, \Sigma \frac{\epsilon_i^2}{\sigma_1^2+\sigma_2^2})$$

and where ϵ_i denotes the components of ϵ, σ_2 and σ_2 are the standard deviations of A_1 and A_2, and the product and sum are taken over the match window. (Very often, the the variances σ_1^2 and σ_2^2 can be considered to be constant. In this case, the summation can be reduced to the sum of the squares of the differences over the march window, with the sum of the two variances factored out.) Thus,

$$P(x_m,y_m=x_c,y_c \mid \epsilon) = k\, w$$

So far, the derivation is quite usual. If we simply wanted to maximize P (for the maximum likelihood solution), we would minimize the above sum (that is, use a weighted least-squares solution). However, because of the fluctations in w caused by the presence of noise in both images, the peak of P in general differs from the center of the distribution of P in a random way due to the random nature of the errors.

Therefore, we define the optimum estimate of the matching position to be the mathematical expectation of x_m,y_m according to the above probability distribution. Thus, letting (x_0,y_0) represent this optimum estimate, we have

$$x_0 = \frac{\Sigma\, w\, x_m}{\Sigma\, w}$$

$$y_0 = \frac{\Sigma\, w\, y_m}{\Sigma\, w}$$

where the sums are taken over the search window. The variances and covariance of x_0 and y_0 are given by the second moments of the distribution around the expected values:

$$\sigma_x^2 = \frac{\Sigma\, w\, x_m^2}{\Sigma\, w} - x_0^2$$

$$\sigma_y^2 = \frac{\Sigma\, w\, y_m^2}{\Sigma\, w} - y_0^2$$

$$\sigma_{xy} = \frac{\Sigma\, w\, x_m\, y_m}{\Sigma\, w} - x_0\, y_0$$

The covariance matrix of x_0 and y_0 consists of σ_x^2 and σ_y^2 on the main diagonal and σ_{xy} on both sides off the diagonal.

It might appear that the above analysis is not correct because of the fact that certain combinations of errors at each point of each picture are possible for more than one match position, and the probability of these combinations is split up among these match positions. However, this fact does not influence the results, as can be seen from the following reasoning. The possible errors at each point of each picture form a multidimensional space. When a particular match position is chosen, a lower-dimensioned subspace of this space is selected, in order to be consistent with the measured brightness values. When another match is chosen, a different subspace is selected. These two subspaces in general intersect, if at all, in a subspace of an even lower number of dimensions. Thus the hypervolume (in the higher subspace) of this lower subspace is zero. Therefore, the fact that the two subspaces intersect does not change the computed probabilities.

Now suppose that the standard deviations σ_1 and σ_2 are not known. It is possible to estimate them (actually, the sum of their squares, which is what is needed in the equation for w) from the data if it is assumed that they are constant, that is, the noise does not vary across the pictures. Let v equal the constant value of $\sigma_1^2 + \sigma_2^2$. Then $\epsilon \cdot \epsilon / w_m^2$ (the mean square value of the components of ϵ) is an estimate for v, where $\cdot$ denotes the vector dot product. However, this value is different for each possible match position x_m,y_m. The method used to obtain the best value for v is to average all of these values for v, weighted by the probability for each match position $p(x_m,y_m = x_c,y_c \mid \epsilon) = w$. Thus a preliminary variance estimate is computed by

$$v' = \frac{\Sigma\, w\, \epsilon \cdot \epsilon}{w_m^2\, \Sigma\, w}$$

where the sums are taken over the search window. However, this averaging process introduces a bias because of the statistical tendency for the smaller values to have the greater weights. It can be shown that this effect causes the estimate of variance to be too small by a ratio that can be anywhere from .5 to 1. Therefore, an empirically determined approximate correction factor is applied to the variance estimate as follows:

$$v = \frac{v'}{1 - 0.5\,(1 - \frac{u}{v'})^{0.3}}$$

where u is the minimum value of $\epsilon \cdot \epsilon / w_m^2$ over the search window. Since the computation of w requires the value of $\sigma_1^2 + \sigma_2^2$ $(= v)$, the above process is iterative.

An estimate of an upper limit to the variance is also computed from the high-frequency content of the pictures. First,

$$\upsilon = \frac{[A(x-1,y) + A(x+1,y) + A(x,y-1) + A(x,y+1) - 4\,A(x,y)]^2}{20}$$

Then υ is averaged over an appropriate local window and the results for the two pictures are added together to form the estimate of the upper limit of v.

The overall variance estimate used in the above equations is obtained by an appropriate weighted combination of the *a priori* given value, the derived value, and the computed upper limit.

The probability of a correct match is computed by comparing the derived variance to the *a priori* variance and the upper limit (high-frequency variance) by means of F-tests.

Because of the finite window size, the computed covariance matrix will be an under-estimate. An approximate correction for this effect is made by computing the eigenvalues and eigenvectors of the covariance matrix, applying a correction to the eigenvalues, and then reconstructing the covariance matrix from the eigenvalues and eigenvectors.

The above computations assume that the shift between the two pictures is always an *integer number of pixels*. In cases where the correlation peak is broad, the smoothing process inherent in the moment computation for x_0, y_0, σ_x^2, σ_y^2, and σ_{xy} cause a reasonable interpolation to be performed if the correct answer lies between pixels. However, when the correlation peak is sharp, this will not happen, and the answer will tend towards the nearest pixel to the correct best match. This is not particularly serious insofar as it affects the position estimate, but it can have a serious effect on the probability estimate. This is because the ϵ vector should be much smaller at the correctly interpolated point than it is at the nearest pixel, because of the sharp peak. Therefore, the probability may come out much too small, indicating a bad match, whereas the match is really good but lies between pixels. To overcome this deficiency, linear interpolation adjustments are made to the variance and probability, and the covariance matrix is augmented to allow for interpolation error.

Since there may be changes in brightness and contrast between the two pictures of the stereo pair, the correlator can adjust a bias and scale factor relating the brightness values in the two pictures. This requires modifying the mathematics given above. Instead of actually using the sum of squares of differences $\Sigma\, \epsilon_i^2$, in the above equations, the moment about the principle axis of the function relating the two sets of brightness values is used. However, the sum of the squares of the differences is still the main ingredient in this computation. Included in this computation are *a priori* weights on the given values of brightness bias and scale factor (contrast). Thus the bias and scale factor can be constrained according to the amount of knowledge about them from other sources, if any.

As stated above, when the variance is assumed to be constant, a major portion of the computation is the sum of squares of differences $\Sigma\, \epsilon_i^2$. This are computed by a very efficiently coded method developed by Moravec [4]. Its inner loop (each term of the summation) requires about one microsecond on the PDP KL10.

Searching for Stereo Matches

Once the stereo camera model is known, the search for matching points in the two pictures is greatly constrained. A point in Picture 1 corresponds to a ray in space, which, when projected into Picture 2, becomes a line segment terminating at the point corresponding to an infinite distance along the ray. Furthermore, by first trying a match with approximately the same stereo disparity as neighboring points that already have been matched, the search can be eliminated for many points. One criterion for deciding when to accept this tentative match is the probability value returned by the high-resolution correlator. Also, when a search is made, the likeliest correct match is indicated by the highest probability value.

The method used here is similar in some ways to matching techniques used by others (for example, Quam [5] and Hannah [2]). However, there is no region growing in the sense of Hannah, since the equivalent operations are left until later in the processing. Instead, the stereo disparities are allowed to vary in an arbitrary way over the picture, subject to some local constraints discussed later. Furthermore, the acceptance of matches is guided by the probability values. Also, even in areas of low information content, the noise suppression ability of the high-resolution correlator often allows useful results to be obtained. If the content is too low, the correlator indicates this fact by producing very large values for the standard deviations of the two position coordinates. When this happens, the searching can be inhibited to save computer time, but even if this is not done, the results are still as good as the standard deviations indicate. (Actually, the correct test to indicate no useful information is to see if both eigenvalues of the covariance matrix are large. Both standard deviations might be large, but if only one eigenvalue is large, an accurate distance can still be computed for this point unless the corresponding eigenvector is almost parallel to the projected line segment.)

The method currently used is approximately as follows:

1. Divide Picture 1 into square windows, denoted here as "areas", the center of each of which is considered to be a point to be matched to the center of a similar area in Picture 2 in the following steps. (These areas normally would be equal in size to the match window of the high-resolution correlator.)

2. Select a set of starting areas. (Currently a column near the edge of the picture is used, but this will soon be changed to the points which were produced by the interest operator and binary-search correlator and were not rejected by the camera model solver.)

3. Try areas adjacent (including diagonally adjacent) to areas already tried, where possible working in the direction of the projected line segments in Picture 2 towards the infinity points.

4. If there are at least two already matched areas adjacent to the area in question and the disparities of all adjacent matched areas agree within a tolerance, apply the high-resolution correlator with the search window centered on the position corresponding to the average disparity of these neighbors. Otherwise, go to 6.

5. If the probability returned by the correlator in step 4 is greater than 0.1, accept this match and go to 8.

6. Starting at the infinity point, search along the projected line segment in Picture 2, applying the search window of the high-resolution correlator at points with a spacing of half of the search window width, but not at previously matched areas.

7. Of those matches found in step 6, select the one for which the correlator returned the highest probability. If this probability is greater than 0.1 and at least one neighboring area (including these tentative matches) agrees in disparity and has a probability greater than 0.01, or vice versa, accept this match. Otherwise, of those matches found in step 6 with probability greater then 0.1, if any, accept the one whose disparity agrees most closely with its neighbors, if within the tolerance.

8. When the current group of areas being tried is exhausted, go to 3. If there are no areas left, finish.

Some improvements can be made to this algorithm in the future. For example, another pass can be made over the data to clean things up, utilizing the fact that most areas have more matched neighbors than they did when things were progressing in a basically one-directional manner. Another possibility is to change step 7 in the following way. The best match from those found in step 6 would not be selected immediately. Instead, all of the potential matches with sufficiently high probability would be saved until the entire picture had been processed. Then a cooperative algorithm similar to that discussed by Marr and Poggio [3] could be used to choose the best matches. This should produce more reliable matches, but with a large increase in computation time.

Ground Surface Finder

Once the three-dimensional positions of a large number of points in an outdoor scene have been determined, it is desired to determine which points are on the ground and which are on objects above the ground. By taking a sufficiently small portion of the scene the ground can be approximated by a simple surface whose equation can be determined, and the points which lie above this surface by more than an appropriate tolerance can be assumed to be on objects above the ground.

Such a procedure has been written, which assumes in general that the ground surface is a two-dimensional second degree polynomial. However, weights can be given to *a priori* values of the polynomial coefficients, to incorporate any existing knowledge about the ground surface into the solution. For example, the second degree terms can be weighted out of the solution altogether, so that the ground surface reduces to a plane.

To determine a ground surface from a given set of data, a set of criteria which define what is meant by a good ground surface is needed. These include the number of points within tolerance of the surface (the more the better), the number of points which lie beyond tolerance below the surface (the fewer the better, since these would be due to errors such as mismatched points in a stereo pair), and the closeness of the surface coefficients to the *a priori* values. Note that the number of points above the surface does not matter (other than that it detracts from the number within the surface), because many points can be on objects above the ground. A score for any tentative solution is computed based on these criteria, and the solution with the highest score is assumed to be correct, although a solution with a lower score can be selected by a higher level procedure using more global criteria. The scoring function currently used is

$$V = \frac{N \cdot m}{N + n - 2m} - \frac{B^2}{b^2 + Bb} - \sum_i \left(\frac{c_i - c_i'}{3\sigma_i}\right)^2$$

where N is the number of points within tolerance of the surface (these points were used to determine the surface by a least-squares fit), n is the *a priori* expected number of points in the surface, B is the number of points below the surface by more than the tolerance, b is the *a priori* approximate maximum number of points below the surface, the c_i are the coefficients of the fitted surface, c_i' are their *a priori* values, σ_i are the standard deviations of these *a priori* values, and m is the number of these coefficients which were adjusted.

Finding the best solution (according to the scoring function) out of all of the possible solutions is a search problem. What is needed is a method which will be likely to find the correct solution without requiring huge amounts of computer time. The method used uses some heuristics to lead the search to the desired solution. Its main points can be described briefly as follows.

First, a least-squares solution is done using all of the points. This fit is saved for refinement leading to one tentative solution. Then all points within tolerance of this fit or too low, but not less than half of the points used in this fit, are selected, and another least-squares fit is done on these points and saved. This process repeats until there are too few points left. (This portion of the algorithm drives downward to find the low surfaces, even though there may a large amount of clutter above them.)

The refinement of each of the above fits is done as follows. The standard deviation of the points used in the fit about the fitted surface is computed. Then all points within one standard deviation (or within the original tolerance) of the surface are used in a new least-squares fit. This process continues until it stabilizes, in which case the score of the result is computed, or until there are too few points in the solution. (This portion of the algorithm rejects erroneous points and some clutter, in order to find well-defined surfaces.)

Results

Figure 1 shows a stereo pair of photographs taken from positions approximately 1.8 feet apart in a parking lot. Each digitized picture is 270 pixels wide and 240 pixels high.

Figure 2 shows the points found in the left picture by the interest operator and the corresponding points (using the same arbitrary symbols) matched in the right picture by the binary search correlator. The points encircled were rejected because of low probability (<0.1) estimated by the high-resolution correlator. The points in squares were rejected by the editing in the camera model solver. The remaining points then determined the camera model solution.

Figure 3 shows the matches produced by the searching algorithm, constrained by this camera model, using the high-resolution correlator (with eight-by-eight windows). Notice that the algorithm made several incorrect matches, particularly in the left foreground. This is a result of the fact that there was very little contrast in the texture on the pavement, resulting in a low signal-to-noise ratio. Nevertheless, there are a sufficient number of correct matches so that the later stages of processing are not bothered by these errors.

The left side of Figure 4 shows the component of distance (in feet) parallel to the optical axis, computed from the matches for all points that the algorithm matched, superimposed on the left picture. (Single characters are use for these plots, with 0 through 9, A through B, and a through b representing 0 through 61. The number sign represents values from 62 to 100, and the infinity symbol represents everything greater than 100. The right side of Figure 4 shows the relative standard deviation computed for these points, to the nearest foot. (The relative standard deviation indicates only those errors which tend to differ for nearby points. The total standard deviation is larger and indicates the absolute accuracy of the distances.)

Figure 5 shows the results of transforming the three-dimensional information represented by the distances into an approximately horizontal coordinate system. The Camera 1 position is at the bottom center of the figure, looking towards the top. Plotted are the heights above the reference plane in feet, using the single characters described above.

The portion of the data between ranges of 10 feet and 50 feet was given to the ground surface finder. The heights of these points above the resulting plane are shown in Figure 6.

Figure 7 is the same as Figure 6, except that it shows only those points with a height of at least two feet. The points above this threshold are on the two vehicles in the pictures, the light poles, and some shrubbery near the light poles, with a few error points.

The results of processing a pair of aerial photographs are shown in Figures 8 through 11. Figure 8 shows the stereo displacements in pixels computed by the searching algorithm for a stereo pair of pictures, superimposed on one of the pictures in the proper positions. (The steps used to obtain the camera model are not shown.) Since the range of heights in the picture is small compared to the height of the camera, these stereo displacements are approximately proportional to heights above an arbitrary plane. Figure 9 shows heights above the ground plane found by the ground surface finder, and Figure 10 shows only those points with heights that would be expected for points on cars. Figure 11 shows heights scaled so that 1 is approximately the height of the hood or fender of a car, 2 is approximately the height of the roof of a car, and 3 through 6 represent heights that might be found on large trucks. In this figure, values of 0 represent points that appear to be on the ground.

Figures 12 through 14 show some results for another pair of aerial photographs with lower resolution. Figure 12 shows heights above an arbitrary plane, and Figure 13 shows heights above the computed ground plane. The points that appeared to be sufficiently above the ground plane to be possibly on buildings were given to the ground surface finder again, with instructions to find a plane parallel to the ground plane, and it found the surface corresponding to the horizontal roof of the large building in the picture. Figure 14 shows the points in the original picture that appear to be on the ground (marked "G") and on the roof (marked "R").

Future Plans

It is planned to use the stereo system in a system for operating an experimental exploring vehicle. Stereo pairs will be taken from various locations and their results will be combined. Points which lie sufficiently above the ground will be clustered into individual objects, and simple size and shape information will be computed for each object. A data structure containing a catalogue of objects, with their locations, sizes, and shapes, and properties of the ground, will be built up as the vehicle moves through its environment. By comparing this information to older portions of the data structure, the vehicle can determine if it is in a previously seen area.

There are several opportunties for the previously described components of the system to cooperate and to pass information back and forth. For example, the high-resolution correlator has several parameters used to give it *a priori* information on the noise level in the data and changes in brightness and contrast between the two pictures. It also produces *a posteriori* estimates of these quantities. These results from early applications of the correlator (for example with the points used to obtain the camera model and not rejected by the camera model solver) can given to the correlator in later applications. Also, when the ground surface finder is given points from a certain portion of the scene, it can be given *a priori* values and weights for the surface, which can be obtained from ground surface solutions for adjacent areas. Furthermore, if its apparently best solution does not agree with those of adjacent areas, an alternate solution from the ground surface finder can be used which agrees better with neighboring areas, even though it may not be as good according to local criteria. All of these features may be implemented in the future.

Other possibilities include an automatic segmenter to produce regions of complicated terrain for the ground surface finder to work on, and the addition of *a priori* knowledge about the environment, including models of objects expected.

Acknowledgements

This paper is adapted from a paper presented at the Fifth International Joint Conference on Artificial Intelligence, Massachusetts Institute of Technology, August, 1977. The research described herein was performed at the Stanford Artificial Intelligence Laboratory under NASA contract NASW-2916 and the Advanced Research Projects Agency Order 2494 contract MDA903-76-C-0206.

References

1. Gennery, D. B., "Stereo-Camera Calibration," Stanford Artificial Intelligence Laboratory Memo in preparation.

2. Hannah, M. J., "Computer Matching of Areas in Stereo Images," Stanford Artificial Intelligence Laboratory Memo AIM-239, Stanford University, July 1974.

3. Marr, D. & Poggio, T., "Cooperative Computation of Stereo Disparity," *Science*, Vol. 194, pp. 283-287, October 15, 1976.

4. Moravec, H. P., "Towards Automatic Visual Obstacle Avoidance," *Fifth International Joint Conference on Artificial Intelligence*, Massachusetts Institute of Technology, August, 1977.

5. Quam, L. H., "Computer Comparison of Pictures," Stanford Artificial Intelligence Laboratory Memo 144, Stanford University, 1968.

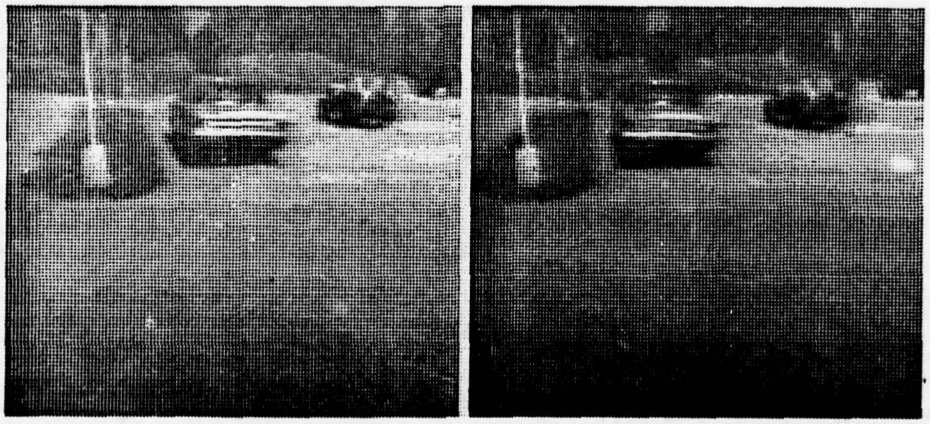


Figure 1. Stereo pair of pictures.

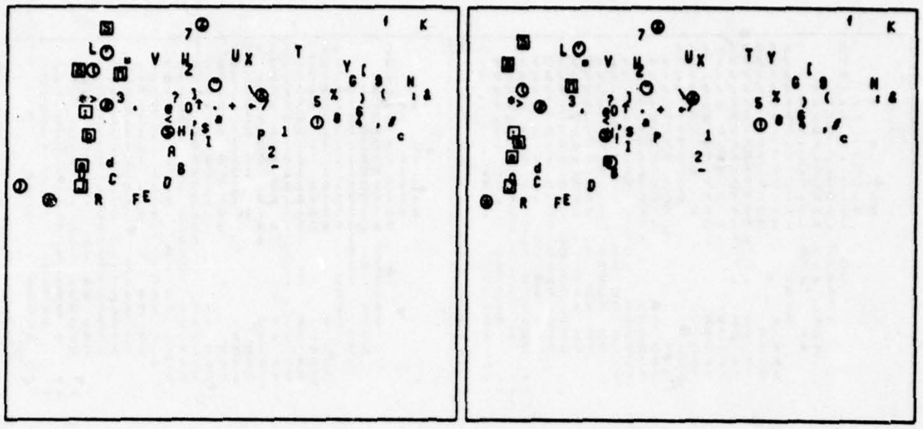


Figure 2. Matching points found by binary-search correlator, rejected by high-resolution correlator (circled), and rejected by camera model (boxed).

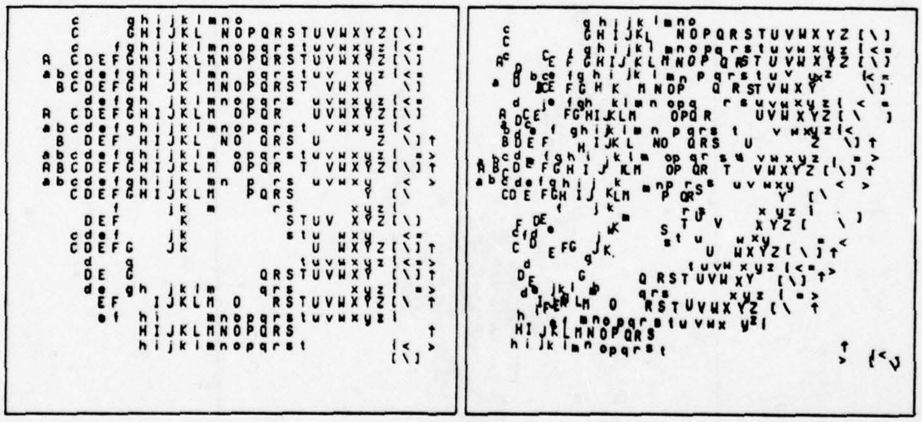


Figure 3. Matching points found by search algorithm using camera model and high-resolution correlator.

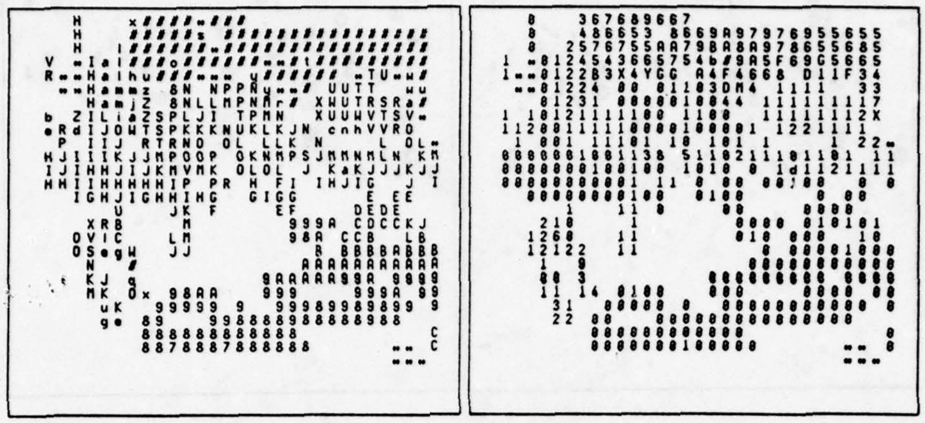


Figure 4. Distances (left) and relative standard deviations (right) to points in left picture of Figure 1. The symbols are defined in the text.

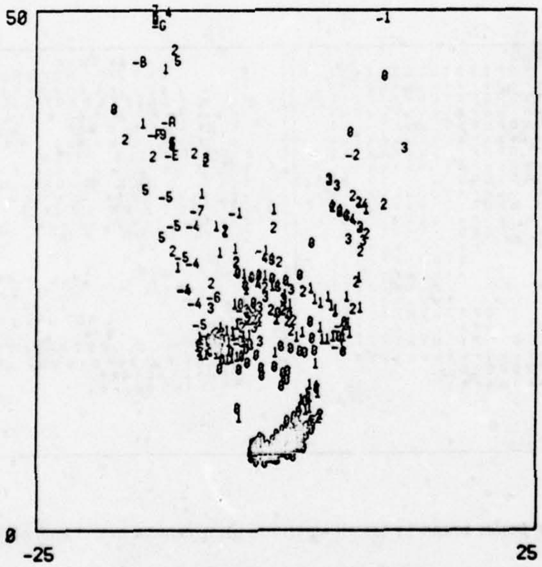


Figure 5. Heights above reference plane.

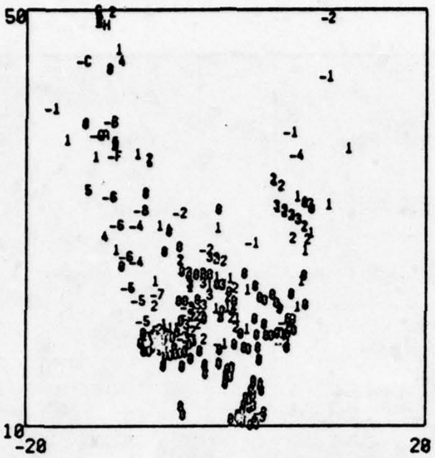


Figure 6. Heights above fitted ground plane.

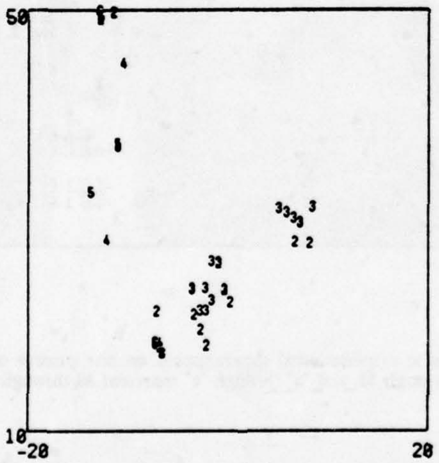


Figure 7. Heights at least two feet above fitted ground plane.

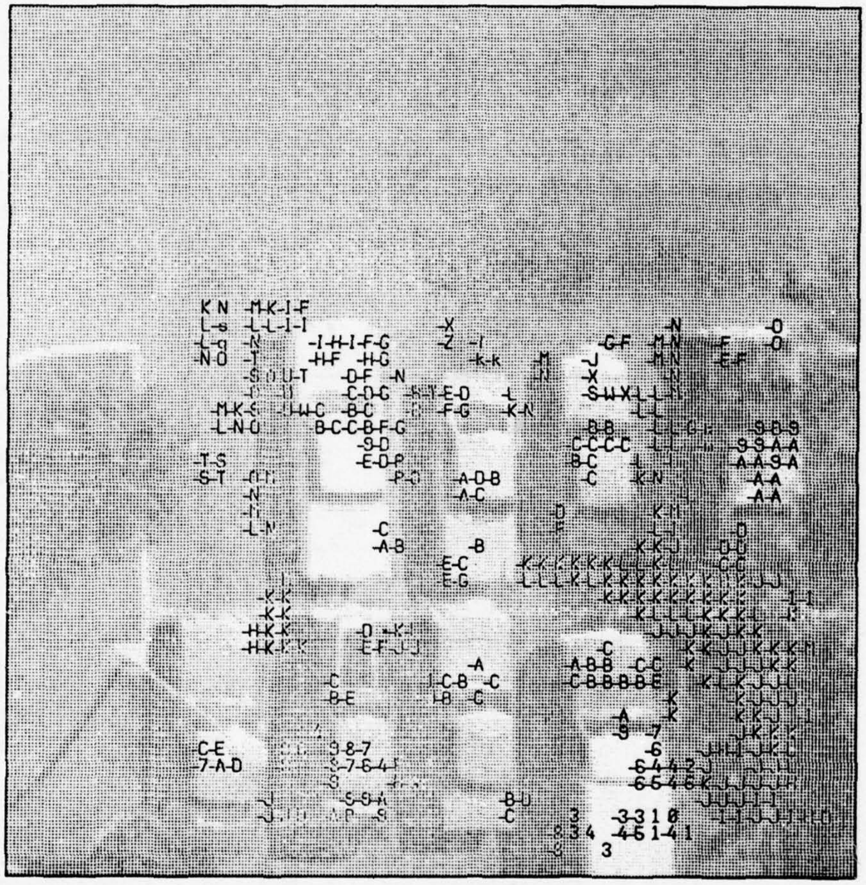


Figure 8. Arbitrary heights (stereo displacements) superimposed on one picture of a stereo pair. ("A" through "Z" represent 10 through 35, and "a" through "z" represent 36 through 61.)

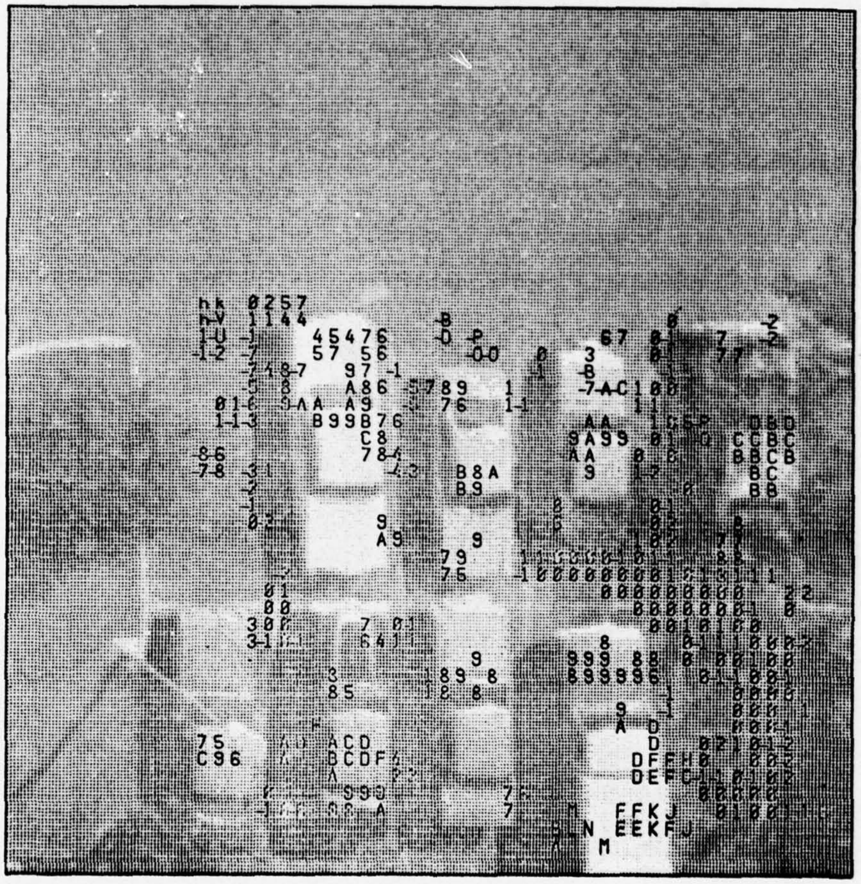


Figure 9. Heights above the fitted ground plane.

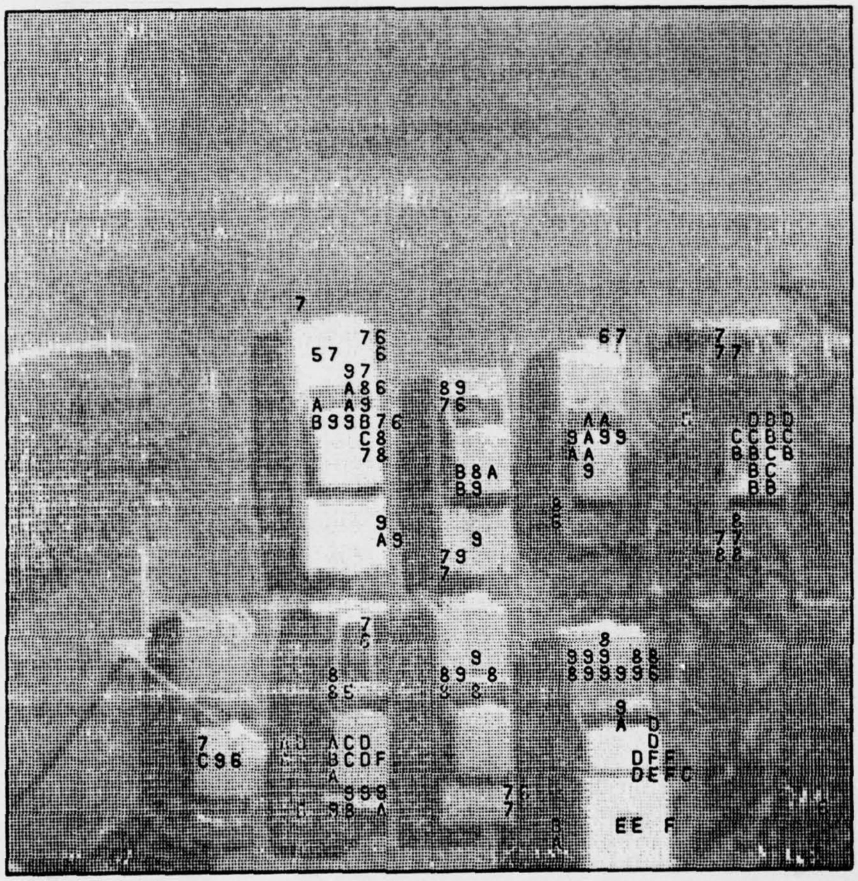


Figure 10. Heights between 5 and 15 units (approximately the range of heights expected for points on cars).

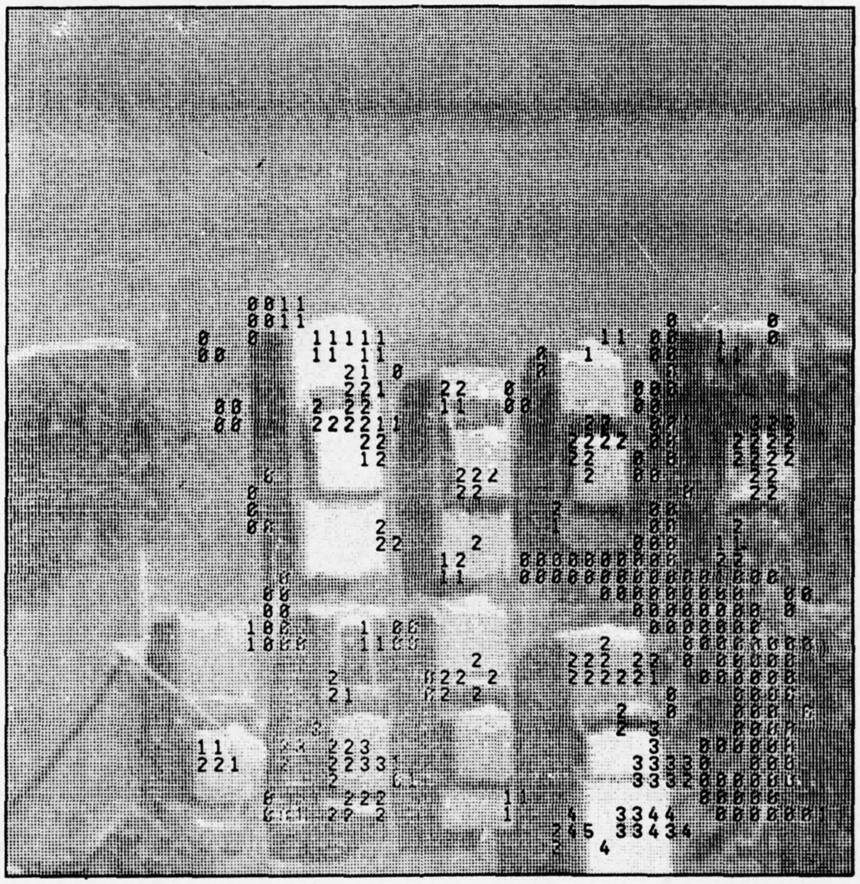


Figure 11. Heights scaled so that the unit is approximately 2.5 feet, showing all points from -0.5 to 6 in this unit.

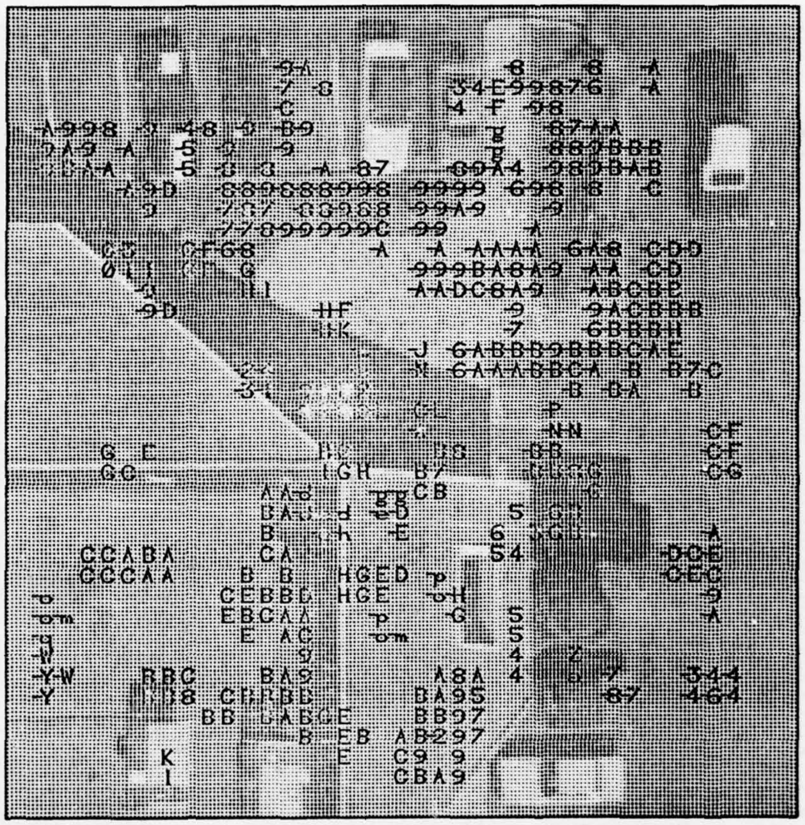


Figure 12. Arbitrary heights (stereo displacements) superimposed on one picture of a stereo pair. ("A" through "Z" represent 10 through 35, and "a" through "z" represent 36 through 61.)

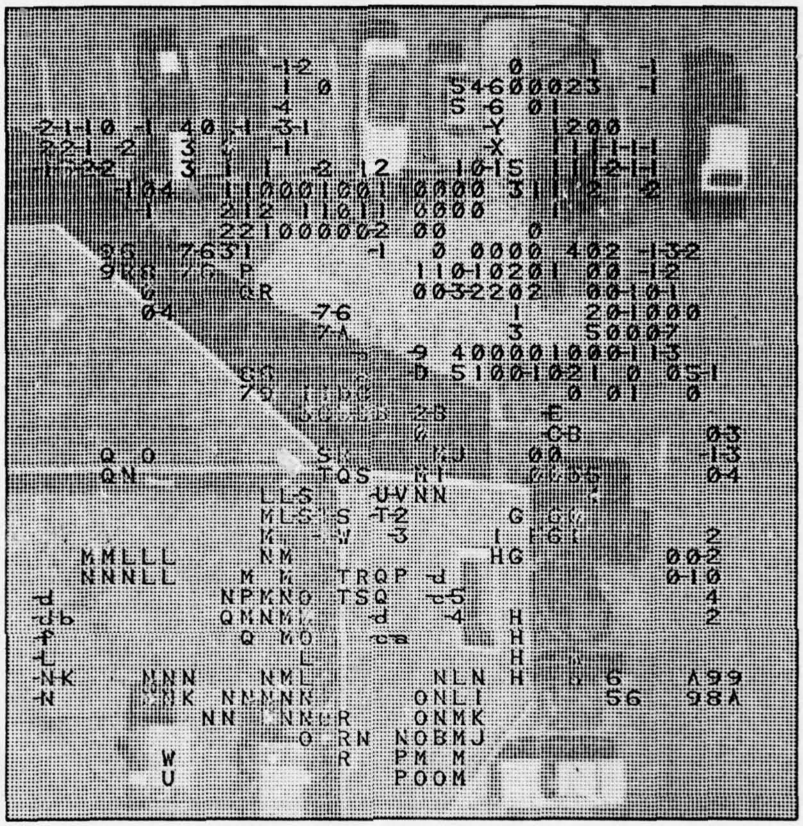


Figure 13. Heights above the fitted ground plane.

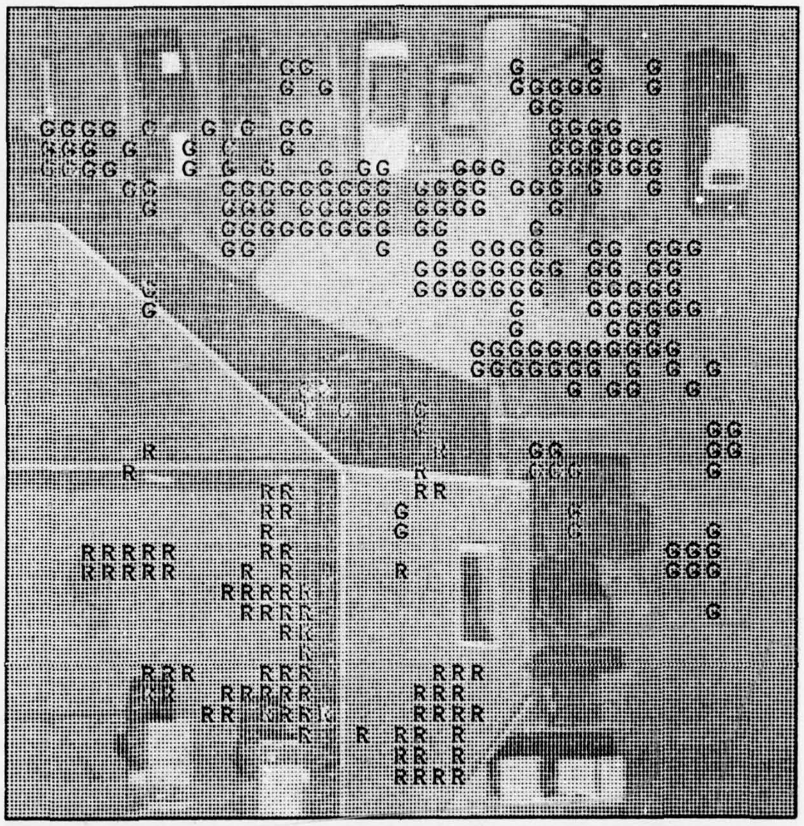


Figure 14. Points within tolerance of ground (G) and within tolerance of horizontal roof plane (R).

The MIDAS Sensor Databasc and its Use in Performance Evaluation

David M. McKeown, Jr. and D. Raj Reddy
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pa. 15213

Abstract

The development of image understanding systems requires the parallel development of software tools to measure and analyze their performance and behavior. This paper briefly describes the design of a sensor database and discusses its use in performance evaluation of segmentations and labeling.

Introduction

Many problems arise in the design of image understanding systems which require large amounts of image data to be readily available for use in a variety of experiments. Such databases must contain not only the images themselves (signal description of the scene) but also symbolic descriptions of the content of the image. A symbolic description is one where a symbolic name is used to represent a collection of points in the image. A symbolic name associates a real-world object or sub-object name with a collection of picture points. Usually one needs a hierarchy of symbolic levels where each level represents a different conceptual abstraction of the image. The maintenance of a hierarchy of these representations is necessary in order to define and analyze the structural relationships between symbolic levels. Both the symbolic and signal descriptions must be organized into a database which allows efficient and accurate representations of images.

While the number and type of representations may vary from task to task it is important that the database provide mechanisms to add or modify representations as necessary. The particular representation paradigm chosen should be applicable to a wide variety of images. The grain of the representation should allow for efficient mappings of the signal (image) into the representation space. MIDAS, a multi-sensor image database system, is currently under development at CMU and attempts to fulfill these design goals (McKeown and Reddy, 1977).

MIDAS Organization

MIDAS is composed of three interactive subsystems (Figure 1). First, the QUERY system can locate images with particular attributes such as *Sensor*, *Scene type*, *Source of image* or *Owner* ie. "all color cityscape scenes processed by Ohlander". The CATLOG system contains functions to insert, delete and modify image representations. The PICPAC system provides general picture modification, analysis, and manipulation procedures.

MIDAS maintains multiple data structures in order to efficiently represent both idealized and experimentally generated scene descriptions. The primary data structure is a set of text files which contain hierarchical symbolic descriptions. Other data structures include a relational database used primarily for interactive query.

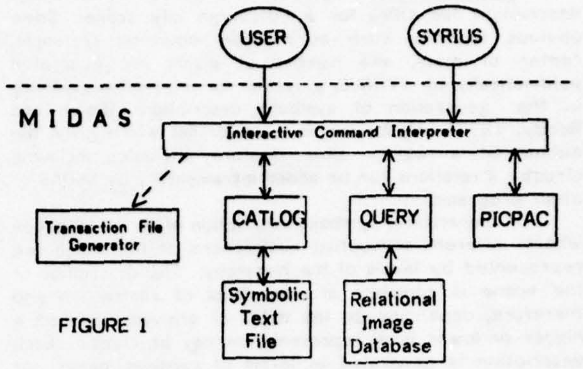

USER
SYRIUS
MIDAS
Interactive Command Interpreter
Transaction File Generator
CATLOG
QUERY
PICPAC
Symbolic Text File
Relational Image Database

FIGURE 1

```
MARKED = HAND-SEGMENTATION
#REGIONS = 14
IDFDAT! = 8/3/77
****************************************
THIS IS THE RUBIN PITTSBURGH CITY SCENE # 12.
SEGMENTATION BY AMY PIERCE.
****************************************
!
REGION# = 1
SYMNAME = SKY
SYMLEVEL = OBJECT
CENTROID = 119,353
MBR = 1,246 1,699
#PIXELS = 165784
AVERINTENSITY = 9
COMPACTNESS =  .0471481
#MASKPOINTS = 12
MASKPOINTS =
1,1  1,699  239,699  237,699  237,687  246,687  244,591
231,2  231,2  231,1  182,1  1,1

REGION# = 2
SYMNAME = BACKGROUND
SYMLEVEL = OBJECT
CENTROID = 273,306
MBR = 231,356 1,687
#PIXELS = 42187
AVERINTENSITY = 7
COMPACTNESS =  .0100491
#MASKPOINTS = 63
MASKPOINTS =
355,1  356,59  304,59  302,59  302,89  298,165  305,161
326,162  326,167  314,168  314,199  298,199  297.209  301.209
303,209  302,219  274,219  274,252  301,252  300,268  268.267
254,267  254,316  287,316  294,316  295,343  295.349  286.349
284,353  256,353  256,360  284,360  284,383  328.383  327.410
309,410  311,456  301,456  301,460  290,460  290,467  290,501
278,501  278,564  294,564  294,607  314,607  321,607  321,662
299,662  294,665  297,675  293,675  289,683  282,683  285,685
290,685  290,687  274,687  246,687  231,1  240,1  355,1

REGION# = 3
SYMNAME = HILTON
SYMLEVEL = OBJECT
CENTROID = 355,104
MBR = 298,417 58,165
#PIXELS = 10423
AVERINTENSITY = 7
COMPACTNESS =  .0554281
#MASKPOINTS = 12
MASKPOINTS =
302,59  410,58  417,83  409,149  409,148  326,148  326,162
326,162  305,161  298,165  302,90  302,59
```

FIGURE 2 Image Description File

Representation. Figure 2 is a portion of an image description file (IDF) for a Pittsburgh city scene. Some obvious features such as minimum bounding rectangle, center of mass, and number of pixels are generated automatically by *SYRIUS*, a system for interactive guidence in the generation of symbolic descriptions (Smith and Reddy, 1977). *Maskpoints* is a vector list which gives the outline of a region. Other feature attributes including structural relations can be added incrementally by MIDAS or other programs.

A hierarchical symbolic description of an image is one where different conceptual abstractions of the image are represented by levels of the hierarchy. The description of the scene is complete at each level of abstraction and therefore, depending on the detail of analysis required, a higher or lower level representation may be chosen. Each description is generated in terms of symbolic names for objects found in the image at that level and the structural relationships between these objects. Structural relations are concepts such as *above, left-of, composed-of* and *vertical*.

A careful study of the capabilities necessary to perform the above, leads us to the following general requirements for our database system. The database should provide uniform access functions to all images and allow the maintenance of partial and alternate representations of images. The notion that the database is only partially complete with respect to the description of each image, requires that the incremental updating of image descriptions be handled in an efficient and consistent manner. Extracted features should be stored with each image description so that the database can be used as a medium for knowledge acquisition. We impose a hierarchy of representations for each image which allows for the efficient evaluation of systems performance at different representational levels. At each representational level there are structures which are given symbolic names and have associated feature sets which were used to derive the structure itself.

```
RELATION  1: (GENERIC-ID,GENERIC-NAME,#BANDS,WHICHBANDS)
RELATION  2: (GENERIC-ID,SENSORTYPE,SCENETYPE,ORIGIN)
RELATION  3: (#DATFILE,DATFILE,BANDTYPE,GENERIC-ID)
RELATION  4: (#DATFILE,#ROW,#COLUMN,BYTESIZE)
RELATION  5: (#DATFILE,#DESCRIP,DAT!,OWNER)
RELATION  6: (#DATFILE,#IDFILE,IDFDAT!)
RELATION  7: (#IDFILE,IDFILE,MARKED,#REGIONS)
RELATION  8: (#IDFILE,REGION-ID#,REGION#)
RELATION  9: (REGION-ID#,SYMLEVEL,SYMNAME)
RELATION 10: (REGION-ID#,REGION-RELS,ASSOCIATED-REGION)
RELATION 11: (REGION-ID#,FEATURE,FEATURE-VALUE)
```

FIGURE 3 MIDAS Relations

Figure 3 give a list of the current working set of relations. Each relation contains several attributes to which particular values can be associated. Some of the attributes are obvious image features: byte size, number of rows, and number of sensor bands. Others are symbolic features: symbolic name and level, and feature values such as size, orientation, and minimum bounding rectangle.

The relational database is generated automatically by MIDAS using a subset of the information contained in the image description files (IDF). Updating the database is performed by modifying the text files which must then be compiled into the relational database. The advantage to this is that the relational database becomes a static data structure where access and search procedures can be tailored to the current structure.

The first argument of each relation is designated as a primary key. When a primary key is bound to a value the relation defines a unique mapping of the primary key to the secondary keys in the relation. Secondary keys, when bound in a relation, return all primary keys that satisfy the relation for all secondary keys specified. Any key, primary or secondary, may take on a *don't care* value during the search and any unbound key will be bound to a value when all bound keys are satisfied.

Applications. There are several areas of research interest which require the availability of organized signal and symbolic data: performance evaluation, error analysis, learning, and knowledge representation. However, the main motivation for this work arises from the belief that uniform representation and organization of images allow researchers to evaluate algorithms and techniques over a wide variety of images without the burden of redeveloping specialized tools for representation. Within our research environment we have several hundred images of various types ranging from earth satellite multi-spectral pictures to electron photomicrographs of ganglia. These images are generaled by a variety of sensors including flying spot scanner, color scanner, side looking radar, electron beam, and LANDSAT multi-spectral scanners. Thus, from a practical standpoint, the database serves an important function if only to keep track of what pictures are available and what previous processing has been performed.

Performance evaluation involves determining how closely a machine generated description matches an idealized description of the image. We shall discuss this in detail in the following sections.

Error analysis is an extension of performance evaluation in that we must be able to describe the nature of errors which occur in automatic scene descriptions. Given mismatched or omitted objects one can begin to localize and diagnose sources of errors in segmentation, feature extraction, and labeling by comparing machine results with the corresponding descriptions within the database. This allows us to localize and identify possible causes of the error.

Programs to learn structural and feature descriptions of objects can be developed given specific exemplars from a variety of images. The automatic learning of symbolic feature primatives (signal to symbol transformations) is essential for progress towards general image understanding systems. MIDAS provides a uniform representation for a large variety of objects occuring in various types of images. In the rest of this paper we shall describe some of the techniques under consideration for performance evaluation.

Performance Evaluation

Performance evaluation of image understanding systems requires the determination of how closely the image interpretation produced by a program matches that of a human. Most systems are developed and debugged on a small set of images and little or no validation of results is performed outside of the working set. Where validation is performed it usually is a subjective analysis rather than along qualitative dimensions. Perhaps this indicates how little we understand about the problem. In this section we will outline our approach to systematic study of performance evaluation. Much of our work is preliminary and our ideas and techniques will surely change with experience, however we believe progress in this area is essential for the long term success of image understanding programs.

Requirements. The first requirement for any type of performance analysis is the ability to generate "the truth" about the image. This *ideal scene description* (ISD) corresponds to what we feel our programs should produce as their output. The ISD comes in many flavors; since symbolic representations are possible at many levels of abstraction, performance of the system can be measured independently at each level. Each level then has its own ISD which must be represented and evaluated in order to measure the goodness of knowledge sources used in image interpretation. For example, at the scene level it might be acceptable if the scene was correctly identified as an office scene, even though some objects within the scene were incorrectly labeled. At a lower level, the evaluation of edge position or region boundaries would be necessary and the ISD would be stated in terms of these features. The evaluation of misplaced, omitted, or added boundaries would be appropriate to provide performance data at these levels.

Problems. There are several problems which are encountered when one attempts to compare scene ISDs with *machine generated descriptions* (MGD). First, our representations are not always correct. There must be facilities to refine and continuously update these idealized descriptions. These representations are created using *SYRIUS*, a system for interactive guidance in the generation of symbolic descriptions (Smith and Reddy, 1977). *SYRIUS* provides convenient means for building ISDs, creating or modifying segmentation masks, and extracting and cataloging attributed features in conjunction with MIDAS. Second, it is not clear that there is only one correct interpretation. Often alternate interpretations of image feature descriptions (especially at the lower levels) are equally acceptable.

Methods. Even given accurate representations, the comparisons and measurements that we wish to make are not straightforward. At each level we must understand what changes are allowable or relevant and determine what knowledge can be brought to bear to aid in the analysis. For example, at the segmental level there are several phenomena to be accounted for. First, where an edge or boundary exists in the ISD, the distance of the same edge in the MGD must be computed and scored. In addition, the frequency of omitted and extra edges in the MGD must be tabulated and scored. The success of a MGD then should be measured as a function of the frequency of extra and omitted edges and a figure of merit for the distance between edges in the ISD and MGD. The implication here is that the ISD representation must allow for uncertainty in the position of edges. This distance is the area between the ISD edge and its counterpart in the MGD.

```
I   xo  I
 I   •   I
 I   •   I
 I   x o I
  I   xo   I
  I   xo   I
  I   •    I       xxxx  Idealized edge
   I   xo   I
    I   •    I     I  I  Uncertainty range
    I  ox    I
   I o x    I      oooo  Realized edge
  I o x    I
  Io  x    I       ••••  Realized matches idealized
 I  ox    I
 I   •    I
```

FIGURE 4 Edge distance metric

Figure 4 illustrates the use of uncertainty intervals in the ISD edge descriptions. No penalty is imposed while the MGD edge remains within the interval, however its distance is measured and scored. If multiple edge points are found within the interval they are scored as extra edges. The lack of an edge within the ISD interval counts as a missing edge.

Labeling accuracy can be measured by comparing, on a point by point basis, the primitive picture element (PPE) (Rubin and Reddy, 1977) assigned in the MGD to that in the ISD. Typical analysis would include the percent of pixels that were correctly labeled and a confusion matrix. A confusion matrix tabulates the frequency at which one PPE is mislaboled as another. This matrix can be used to tune the feature descriptors which define the PPE or indicate when PPE's should be split or grouped into new classes. Some preliminary results using data produced by the ARGOS system (Rubin, 1977) are given in Figure 5. The frequency distribution of PPEs for both training and labeling phases are shown along with the confusion matrix. We plan to improve this analysis by measuring the feature distances for PPEs which are frequently confused.

At the region level it is also possible to use simple techniques such as pixel counting to determine how well the image was partitioned into regions. We propose the following procedure for determining the relative goodness of alternative region partitions. It is necessary to register regions in the MGD with those in the ISD and count the number of pixels covered and missed by the MGD regions. Registration is performed by calculating the number of pixels which overlap between the ISD and MGD regions on a pairwise basis. An initial covering of MGD regions onto ISD regions is made such that the the mapping is 1 to 1 and each of the ISD regions has been assigned an MGD region which covers the greatest area of the ISD region. Any remaining MGD regions are mapped so that they maximize the number of ISD region points covered. Once all MGD regions have been registered the number of pixels covered and missed can be counted. A figure of merit can be determined from the percent of area correctly covered and the number of missing or extra regions in the MGD.

Figure 6 gives an example of this procedure for two alternative machine descriptions: MGD I and MGD II. The shaded area in the registered MGDs corresponds to the number of pixels which were missed. The ratio of missed to covered pixels is one relative measure of goodness. Using this criteria MGD I (40/256) is a better region description that MGD II (56/256). However note that MGD I generated two extra regions while MGD II produced only one more than the ISD. Penalties must be factored into the scoring to handle cases where a MGD with a large number of extra regions covers the ISD better than one with the exact number of regions.

Conclusions

We have illustrated how the MIDAS sensor database can be used in the performance evaluation domain. The examples described are the beginning of a set of performance evaluation tools currently being developed.

References

McKeown, D. M. and Reddy, D. R. (1977). "A Hierarchical Symbolic Representation for an Image Database," Proceedings of *IEEE Workshop on Picture Data Description and Management*, April, 1977.

Rubin, S. M. and Reddy, R. (1977). "The LOCUS Model of Search and its Use in Image Interpretation," Proceedings of *5th IJCAI*, August, 1977

Rubin, S. M. (1977). "The ARGOS Image Understanding System" (thesis in preparation) Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pa.

Smith, D. and Reddy, D. R. (1977). "Interactive Generation of Symbolic Representations of Images," unpublished report, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA.

```
LABELING FREQUENCY COUNT             TOTAL NUMBER LABELED = 7500
PPE -  Not Assigned          Freq: 0       -
PPE 1  HILTON                Freq: 792     10%
PPE 2  GATEWAY 1             Freq: 847     11%
PPE 3  GATEWAY 2             Freq: 409     5%
PPE 4  GATEWAY 3             Freq: 170     2%
PPE 5  GATEWAY 4             Freq: 0       -
PPE 6  JENKINS ARCADE        Freq: 2       -
PPE 7  HORNES                Freq: 122     1%
PPE 8  PITTSBURGH PRESS      Freq: 594     7%
PPE 9  STATE OFFICE          Freq: 320     4%
PPE 10 RIVER                 Freq: 12      -
PPE 11 ONE GATEWAY           Freq: 0       -
PPE 12 GATEWAY TOWERS        Freq: 357     4%
PPE 13 ROAD                  Freq: 461     6%
PPE 14 PARK                  Freq: 767     10%
PPE 15 MOUNTAINS             Freq: 1526    20%
PPE 16 SKY                   Freq: 1121    14%
```

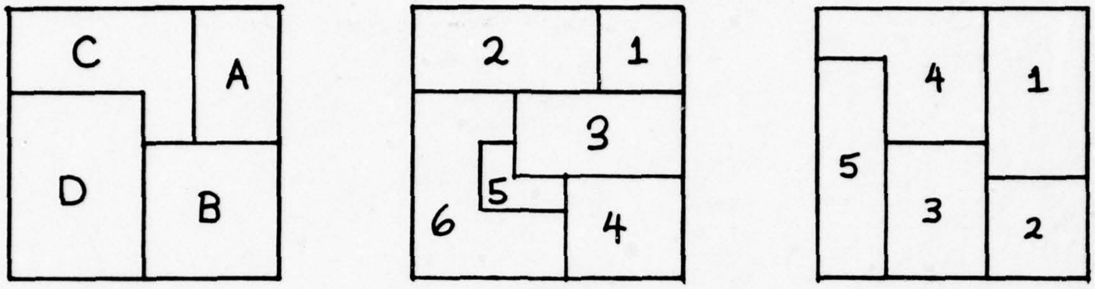
```
TRAINING FREQUENCY COUNTS             TOTAL NUMBER OF TRAINING
PPE -  Not Assigned          Freq: 2856    61%        POINTS = 4644
PPE 1  HILTON                Freq: 451     9%
PPE 2  GATEWAY 1             Freq: 196     4%
PPE 3  GATEWAY 2             Freq: 532     11%
PPE 4  GATEWAY 3             Freq: 149     3%
PPE 5  CATEWAY 4             Freq: 0       -
PPE 6  JENKINS ARCADE        Freq: 17      -
PPE 7  HORNES                Freq: 103     2%
PPE 8  PITTSBURGH PRESS      Freq: 388     8%
PPE 9  STATE OFFICE          Freq: 149     3%
PPE 10 RIVER                 Freq: 72      1%
PPE 11 ONE GATEWAY           Freq: 0       -
PPE 12 GATEWAY TOWERS        Freq: 559     12%
PPE 13 ROAD                  Freq: 60      1%
PPE 14 PARK                  Freq: 234     5%
PPE 15 MOUNTAINS             Freq: 697     15%
PPE 16 SKY                   Freq: 1037    22%
```

```
CONFUSION MATRIX  (FREQUENCY)

          PPE -   1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16   LABELING
          -   -   353  201  105  58   -    -    47   201  87   -    -    54   350  527  797  76   2856
          1   -   424  1    1    10   -    -    -    11   1    -    -    -    -    2    1    -    451
          2   -   -    121  -    -    -    -    4    -    63   -    -    -    -    -    8    -    196
          3   -   -    279  214  -    -    -    -    -    27   -    -    -    -    -    12   -    532
          4   -   7    32   7    97   -    -    -    -    -    -    -    -    -    -    6    -    149
          5   -   -    -    -    -    -    -    -    -    -    -    -    -    -    -    -    -    -
          6   -   -    -    -    -    -    2    15   -    -    -    -    -    -    -    -    -    17
          7   -   -    15   -    -    -    -    54   -    34   -    -    -    -    -    -    -    103
          8   -   5    -    -    -    -    -    2    379  2    -    -    -    -    -    -    -    388
          9   -   -    38   -    -    -    -    -    3    106  -    -    -    -    1    1    -    149
          10  -   1    -    11   -    -    -    -    -    -    12   -    -    48   -    -    -    72
          11  -   -    -    -    -    -    -    -    -    -    -    -    -    -    -    -    -    -
          12  -   2    158  71   5    -    -    -    -    -    -    -    303  3    3    14   -    559
          13  -   -    -    -    -    -    -    -    -    -    -    -    -    60   -    -    -    60
          14  -   -    -    -    -    -    -    -    -    -    -    -    -    -    234  -    -    234
          15  -   -    2    -    -    -    -    -    -    -    -    -    -    -    -    687  8    697
          16  -   -    -    -    -    -    -    -    -    -    -    -    -    -    -    -    1037 1037
TRAINING      -   792  847  409  170  -    2    122  594  320  12   -    357  461  767  1526 1121
TOTALS
```

```
Total number of Pixels in picture: 7500
Number of Pixels assigned during Training: 4644        Percentage of all Pixels 61%
Number of Pixels assigned during Recognition: 7500     Percentage of all Pixels 100%
Number of Pixels correctly labeled: 3730               Percentage of all Labeled Pixels 49%
Number of Pixels incorrectly labeled: 914              Percentage of all Labeled Pixels 12%
Number of Pixels of unknown correctness: 2856          Percentage of all Labeled Pixels 38%
```

FIGURE 5



ISD

MGD I
Two extra regions

MGD II
One extra region

REGION OVERLAP

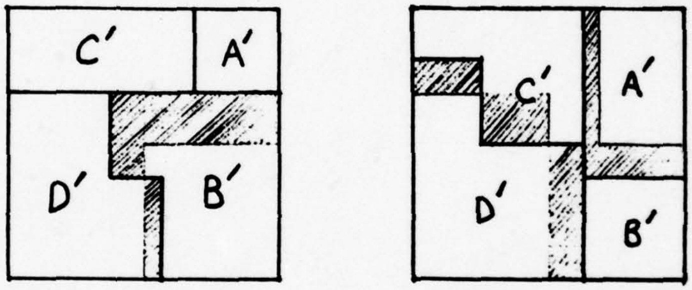
MGD I	MGD II
A1 25	A1 48
A3 15	
B4 42	B2 36
B3 16	B3 16
B6 4	B1 12
B5 2	
C2 55	C4 48
C3 9	C1 8
	C5 8
D6 66	D5 44
D5 12	D3 32
D3 18	D4 44

BEST MAPPINGS

MGD I	MGD II
C→2	A→1
D→6	C→4
B→4	B→2
A→1	D→5

BEST COVERS

MGD I	MGD II
C→2	A→1
D→6+5	C→4
B→4+3	B→2
A→1	D→5+3



REGISTERED MGD I
Missed 40

REGISTERED MGD II
Missed 56

Shaded area indicated pixels which were incorrectly covered

FIGURE 6

SESSION III

RECOGNITION

LOCATING STRUCTURES IN AERIAL IMAGES*

Ramakant Nevatia
and
Keith Price

Image Processing Institute
University of Southern California
Los Angeles, California 90007

ABSTRACT

Analysis of aerial images and location of small structures is a complex task. However, larger objects can be conveniently located by using segmentation techniques best suited for their component extraction. For example, the edge techniques are suitable for extraction of roads and the region techniques for lakes and rivers. Specific objects of interest may be located by their relationships with these more easily extracted objects. Initial results of work in programs are presented.

INTRODUCTION

Analysis of aerial images is, in general, a complex task. The reasons for such complexities are many and varied. A prime cause is the presence of texture which causes difficulties for the low level processes such as edge detection and segmentation. Another source of difficulty is that the desired objects and structures may be small compared to the size of a complete image. A detailed analysis of a complete high resolution aerial image is generally prohibitive because of the computational costs.

For many applications, however, a complete and general analysis is unnecessary. Specific structures of interest may have special properties, known a priori, that allow for their easy extraction. The problem of searching for small structures is helped by locating them by their spatial relationships to larger, more easily located structures.

In previous work, we compared two segmentation techniques, the edge based and the region based methods, and concluded that one or the other may be suited for extraction of particular types of structures [1]. This describes our initial attempts to use both techniques, taking advantage of their respective strong points.

PROBLEM DESCRIPTION AND REPRESENTATION

The problem approached is that of finding user specified structures in aerial images. The user specifies the properties useful for the location of the desired structure and also of other related structures. (An interactive, question-answer dialog system is being developed to facilitate interaction with a user, see [2].) This amount of a priori knowledge is likely to be available in many applications of guidance and photo-interpretation.

The a priori information is stored as properties of objects and their relationships to each other, and may be viewed as constituting a graph structure with the objects as nodes and relationships as arcs. The properties and relationships will, in general, need to be unrestricted. Currently, an object is described either by a collection of line segments or by its region properties. The segments are described by their length and width. The regions are described by properties such as brightness (color) and simple shape measures (area, perimeter, ratio of area to perimeter squared, elongation, etc.).

The relationships used are those of relative locations of the different objects and the symbolic relationships of left, right, above and below. Other relationships such as symmetry and similarity are obviously useful, but have not been implemented.

Our representation and use of knowledge is similar to that described by Tenenbaum [3]. The principal difference is in Tenenbaum's use of single pixel attributes to uniquely distinguish objects (in a given context). We use object attributes to aid in the segmentation of the image and then use the attributes of larger, segmented parts for recognition.

*This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Wright Patterson Air Force Base under Contract F-33615-76-C-1203 ARPA Order No. 3119.

FEATURE EXTRACTION AND SEGMENTATION

Feature extraction and segmentation is guided by the properties of the desired objects to be extracted. Thus, an edge detection-line finding process is applied to extract desired linear segments (such as roads) and a region segmentor for extracting areas uniform in some property (for example lakes and other bodies of water).

Consider the aerial image shown in figure 1 (the displayed image contains 352 x 352 pixels, an image of twice the resolution is also used in the analysis). Here, an objective may be to locate the dock structure and perhaps some ships in it. As this structure consists of relatively small parts and is complex, it may be easier to extract related structures such as the river, the major highway and the lakes first, and use these to concentrate the search for docks to a smaller area of the image. (We assume such information is supplied by the user. No attempt has been made to automate the strategy generation process, as in [4].)

Edge detection processes are appropriate for the extraction of the desired roads. Figure 2 shows the results of applying a Hueckel edge detector [5] on the image of figure 1 and linking the resulting edge segments in elongated segments [6]. The road is known to be narrow enough that the edges corresponding to it are of the "line" type (as contrasted with a step edge). Restricting the linked edges to be only of line type results in fewer segments (shown in figure 2).

The lakes and parts of the river are conveniently extracted by using the Ohlander-Price Segmentor [7]. It is known that the desired objects are relatively dark and uniform in intensity, and the dark peak in the intensity histogram should be used for segmentation. Figure 3 shows the intensity histogram for this image. The completed segmentation is shown in figure 4.

MATCHING OF SEGMENTS

The next step is to match the derived line segments and regions with a model of the image. This phase of our work is in progress and experimental results are expected to be available soon. Assuming that the derived segments are distinctive enough to be easily distinguished, approximate locating of the dock structures can be predicted. Now, sensitive line detectors should help locate the piers of the dock. (We have found the Hueckel edge detector to be deficient in locating small edges, perhaps because of the large neighborhood size used. Development of more sensitive edge and line detectors is being carried out concurrently, see [8].)

CONCLUSIONS

Some results of processing a complex, aerial image using both the line and the region based techniques have been shown. It appears that the use of simple techniques, specifically suited to particular objects in an image, may allow useful processing of rather complex images. This work is in initial stages of development and the array of segmentation attributes is limited. While it is hoped that the described techniques have general applicability, our experience with real images is, as yet, limited.

REFERENCES

1. R. Nevatia and K. Price, "A Comparison of Some Segmentation Techniques," Proceedings: Image Understanding Workshop, Minneapolis, Minnesota, April 1977, pp. 55-57.

2. K. Price, "An Interactive User System," University of Southern California, USCIPI Report 770, October 1977.

3. J.M. Tenenbaum, "Locating Objects by Their Distinguishing Features in Multisensory Images," Computer Graphics and Image Processing, Vol. 2, No. 3, December 1973.

4. T.D. Garvey and J.M. Tenenbaum, "On the Automatic Generation of Programs for Locating Ojbects in Office Scenes," Proceedings of the Second International Joint Conference on Pattern Recognition, Copenhagen, Denmark, August 1974, pp. 162-168.

5. M. Hueckel, "A Local Operator Which Recognizes Edges and Lines," Journal of the ACM, October 1973, pp. 634-647.

6. R. Nevatia, "Locating Object Boundaries in Textured Environments," IEEE Transactions on Computers, Vol. 25, No. 11, November 1976, pp. 829-832.

7. R. Ohlander, "Analysis of Natural Scenes," Ph.D. Thesis, Department of Computer Science, Carnegie Mellon University, April 1975.

8. P. Chuan, "Development of Edge Detectors for the Extraction of Linear Segments," University of Southern California, USCIPI Report 770, October 1977.

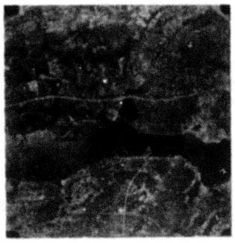


Figure 1. An aerial picture.

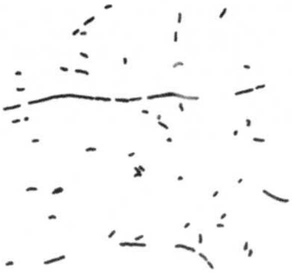


Figure 2. Lines detected in figure 1.

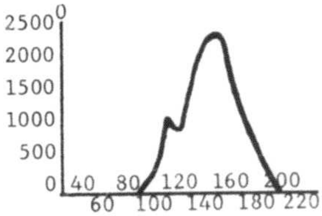


Figure 3. Histogram of the image.

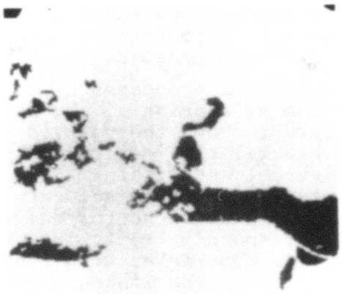


Figure 4. Dark regions detected in figure 1.

3-DIMENSIONAL AIRCRAFT RECOGNITION USING FOURIER DESCRIPTORS

T. Wallace and P. A. Wintz
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

ABSTRACT

The Fourier Descriptor method is a well-known method of describing the shape of a closed figure. Previous work with Fourier Descriptors has suffered from either a loss of shape information, or excessive computation time in comparing an unknown contour to a known one. This paper presents a technique for normalizing Fourier descriptors which retains all shape information, and is computationally efficient.

In recognizing three-dimensional objects, most of the computation time is typically spent in computing distances between an unknown feature vector and a library of feature vectors representing the objects of interest. An interpolation property of Fourier descriptors is described which permits a substantial reduction in the density of projections representing a three-dimensional object. Preliminary experimental results are presented in which this algorithm is applied to recognition of aircraft outlines.

FOURIER DESCRIPTORS

The Fourier Descriptor (FD) is one method of describing the shape of a closed, planar figure. Given a figure in the complex plane, the contour can be traced, yielding a (one-dimensional) complex function of time. If the contour is traced repeatedly, the periodic function which results can be expressed in a Fourier series. The FD of a contour is defined to be this Fourier series.

To implement this method of shape description, it is necessary to sample the contour at a finite number of points. Since the discrete Fourier transform of a sequence gives us the values of the Fourier series coefficients of the sequence, assuming it to be periodic, using an FFT algorithm satisfies the definition above. The computational advantages of the FFT are well known.

Once the Fourier descriptor has been computed, the operations of rotation, scaling, and moving the starting point are easily implemented in the frequency domain by simple arithmetic on the frequency domain coefficients. While shapes may be compared in the space domain, the procedures required to adjust their size and orientation are computationally very expensive. Normally an iterative type of algorithm is employed, which searches for an optimum match between the unknown shape and the reference set.

Granlund's [1] approach to shape information extraction involves defining "Fourier descriptors" by considering products of Fourier series coefficients which are shown to be invariant to position, size, orientation, and starting point factors. This results in an increase in data dimensionality from N to $N^2/2$, without any change in total information. (Since the FFT is a reversible linear transformation, all the shape information is contained in the original N coefficients.) Granlund also computed his FD coefficients using digital or analog integration techniques, which are quite expensive computationally.

Persoon and Fu [2] retained all the shape information inherent in the original contour, and reduced the problem to one of finding an optimum size, orientation, and starting point match between each sample FD and the reference FD. This optimization process was used to check the similarity of each of the possible reference contours. While this method does retain all of the shape information, and guarantees to find an optimum size, orientation, and starting point match, it is still fairly time consuming. In addition, the FDs themselves were computed by an integration process.

Richard and Hemami [4] computed the FDs using an FFT, and then normalized their magnitudes only. They were able to perform magnitude classifications efficiently using this technique, but the true distance measure, including all shape information, required two FFTs for each comparison of an unknown FD to a reference FD.

This paper describes an algorithm which computes FDs efficiently using the FFT, and normalizes the FFT output vector to a standard size, orientation, and starting point before comparing it to any reference FDs. While exact minimization of the distance between two arbitrary shapes is not guaranteed, if the shapes are similar enough to warrant an identification of the unknown shape, the distance found will be very close to the minimum. A general normalization algorithm is presented, and additional theorems are presented for the case of contours possessing bilateral symmetry. The classification problem is also discussed, and relationships between contours and their FDS are investigated.

NORMALIZATION

The frequency domain operations which affect the position, size, orientation, and starting point of the contour follow directly from properties of the DFT. To change the position of a contour, just vary the zero frequency (DC) coefficient of the FD. Adding a complex constant to every point in the time domain representation of a contour is equivalent to adding that value to the DC term of the DFT.

To change the size of the contour, the components of the FD are simply multiplied by a constant. Due to linearity, the inverse transform will have its coordinates multiplied by the same constant.

To rotate the contour in the time domain simply requires multiplying each coordinate by $e^{j\theta}$ where θ is the angle of rotation. Again by linearity, the constant $e(j^{\theta})$ has the same effect when the frequency domain coefficients are multiplied by it.

To see how the contour starting point can be moved in the frequency domain, recall the time shifting property of the DFT. Shifting the starting point of the contour in the time domain corresponds to multiplying the ith frequency coefficient in the frequency domain by $e(j^{iT})$, where T is the fraction of a period through which the starting point is shifted. (As T goes from 0 to 2π, the starting point traverses the whole contour once.)

Given the FD of an arbitrary contour, the normalization procedure requires performing the normalization oprations such that the contour has a standard size, orientation, and starting point. The following method of FD normalization preserves all of the shape information while rejecting noise effectively. In order to reject noise, the coefficients used in the procedure are chosen to have as large magnitudes as possible.

First, we require the phases of the two largest coefficients to be zero. A(1) will always be the largest, with magnitude unity due to the scale normalization procedure which defines that magnitude. Let the second largest coefficient be A(k). (The frequencies of the coefficients produced by an FFT of length N range from -(N/2)+1 to (N/2)) the normalization multiplicity M of coefficient A(k) is defined as:

$$M = |k-1|$$

Thm: The requirement that A(1) and A(k) have zero phase angle can be satisfied by M different orientation/starting point combinations.
Proof: Use the two allowable operations to arrive at one orientation and starting point which gives zero phase for A(1) and A(k). Next use the starting point movement operation (multiplication of the ith coefficient by e^{ijT} to move the starting point once around the entire contour. To accomplish this T must range from 0 to 2π. Now consideer the two cases k positive and k negative. If k is positive, the phases of A(1) and A(k) will coincide at k-1 different starting points. But at each of these starting points, we can use the orientation operation (multiplication of each coefficient by $e(j^{\theta})$ to reduce the phases to zero. Similarly, if k is ngative, the phases of A(1) and A(k) will coincide at 1-k different starting points. Again, the orientation operation can reduce the phases to zero.

Note that if k=2, the orientation and starting point are defined uniquely. In general, however, A(2) will not be the second largest coefficient in magnitude so this ambiguity must be resolved to achieve a general procedure.

The obvious method of solving this problem is to check the phase of a third coefficient A(p) at each of the M possible orientation/starting point combinations and choose the normalization which gives a phase closest to zero for this coefficient. However, this ambiguity-resolving coefficient cannot be chosen arbitrarily. If the normalization multiplicity of coefficient A(p) is the same as that of A(k), or a multiple of it, the phase of A(p) will be the same at each possible normalization! If M for coefficient A(p) (denoted M[p]) is a factor of M[k], or a multiple of a factor of M[k] less than M[k], there is also ambiguity since some of the M possible normalizations will result in identical phases for A(p). If these ambiguous coefficients are removed from consideration, and the unambiguous coefficient with the largest magnitude is used to select one of the M allowable normalizations, a general procedure is obtained.

To briefly review the entire normalization procedure, we start by dividing each coefficient by the magnitude of A(1) to normalize the size of the contour. We find the coefficient of second largest magnitude and compute its normalization multiplicity. We then locate the third largest coefficient suitable for resolving the ambiguity (A(p)) as explained above. The orientation and starting point are adjusted to satisfy the restrictions that A(1) and A(k) are real and positive, and A(p) has phase as close to zero as possible.

This method is quite powerful, but a slight modificaton in the procedure has been found helpful in those cases in which there are two or more coefficients suitable for normalization with almost the same magnitude. It is very unlikely that the magnitudes will be identical, but if they are even close, noise may cause one of them to be used to normalize the test FD, and the other to normalize the unknown FD. To overcome this, the normalizaton coefficients used to normalize the test FD can be supplied to the normalization subroutine directly, rather than having the subroutine compute them.

CLASSIFICATION METHODS

Given two normalized FDs (NFDs), how do we measure their degree of similarity? An appropriate classification method is essential if we are to compare unknown shapes to a test set.

Consider two sampled contours a(i) and b(i), and define the difference c(i) = a(i) - b(i). Evidently if a(i) and b(i) are identical, c(i) is identically zero. If a(i) and b(i) are not identical, the magnitudes of the c(i) coefficients are a reasonable measure of the difference between a(i) and b(i). Now consider the frequency domain vectors corresponding to a(i), b(i), and c(i), denoted a(i), b(i), and c(i). Due to linearity, we have c(i) = a(i) - b(i). Applying Parseval's theorem to the difference vector, we find that the sum of the squares of the differences of the real and imaginary parts of each coefficient of two FDS is proportional to their point by point mean square error in the space domain. The mean square distance measure in the frequency domain is seen to correspond to a reasonable time domain criterion which weights each point equally. In recognizing a contour corrupted by such factors as quantization error or poor photographic resolution, such a criterion seems appropriate. The effectiveness of this classification method is demonstrated by the experiments described below.

Since the closed contour is a continuous function, the Fourier series converges fairly rapidly, as would be expected. M.S. of the M.s. distance between two FDs is due to relatively few coefficients, and the classifications reported below use no more than 30 coefficients.

FD - CONTOUR RELATIONSHIPS

If a FD consists of coefficient A(1) only, with all other coefficients zero, it will transform back to the time domain as a sampled circle. Higher frequency coefficients also transform back as sampled circles, but they traverse the circle a number of times. A(k) will yield a time domain seequence which traces a sampled circle k times in the the counterclockwise direction. A(k) and A(-k) together yield a sampled ellipse, in a manner analogous to the elliptical polarization of electromagnetic theory.

Due to linearlity, a contour in the time domain consists of a sum of the inverse transforms of is FD coefficients. Hence this view of each FD coefficient as a sampled "phasor" yields insight into the relationships between a contour and its FD. A(1) is the fundamental frequency coefficient which is always the largest in magnitude, and is forced to have magnitude unity by the magnitude normalization procedure. It is of interest to describe the figures generated by A(1) and A(k) combined, with all other coefficients zero, since often most of the "energy" of a FD is contained in as few as two coefficients. Interestingly enough, the "normalization multiplicity" M defined above plays a part here, with the contour resulting from nonzero A(1) having M[k]-fold rotational symmetry. Granlund observed that contours with k-fold rotational symmetry consist of components whose frequencies are multiples of k-1. If k is negative, the contours are similar to polygons, and if k is positive, the contours generally are quite round, and appear to be loops superimposed on a circle. If k=-1, the contour is of course a sampled ellipse. Most contours of interest taken from actual photographic data have a negative frequency coefficient as the second largest in magnitude.

Figure 1 shows four contours whose FDs have only two nonzero coefficients completely determine the shape of the figure generated, with the phases only affecting orientation and starting point. Note also that the uniform sampling condition in the time domain is not satisfied when any arbitrary FD is inverse transformed.

Consider now a bilaterally symmetric contour in the time domain. It can be shown [5] that a Fourier Descriptor represents a bilaterally symmetric contour iff the rotation and starting point shift operations can be performed such that the imaginary part of each FD coefficient (except A(0)) is zero.

2-DIMENSIONAL AIRCRAFT RECOGNITION

This method of extracting shape information was experimentally tested on 20 airplane silhouettes which were digitized to two different resolution versions were quite accurate representations of the aircraft, while the low resolution versions showed significant distortion of some of the smaller features such as engines. Using the high resolution contours as a test set, an attempt was made to classify the low resolution contours using this FD algorithm. Using a mean square distance measure, 95% classification accuracy was attained. The aircraft were of four different types. Figures 2 and 3 show high and low resolution contours representing each type. Figure 4 shows the magnitudes of the NFDs computed from the high resolution contours.

The aircraft outlines are approximtely bilaterally symmtric, although quantization error prevents them from being exactly symmetric. The normalization procedure will always yield a NFD whose inverse transform has starting point on the real axis, and whose axis of symmetry coincides with the real axis, given the FD of a bilaterally symmetric contour. Which of the two points at which the axis of symmetry intersects the contour will actually be the starting point depends on the ambiguity-resolving procedure described above. The procedure generally favors the point furthest from the origin of the complex plane, but supplying a selected ambiguity-resolving coefficient to the normalization subroutine can reverse this. In ase both possible starting points are approximately equidistant from the origin, the starting point resIting from normalization is somewhat unpredictable. This is the situation in which it is advisable to check that the unknown FD is normalized using the same ambiguity-resolving coefficient as the test FD.

Since the actual experimental contours investigated were not perfectly bilaterally symmetric, the normalization subroutine did not always result in a starting point which falls on the best estimate of the axis of symmetry. However, since the algorithm was written to reject noise, the starting point was always quite close to the axis of symmetry.

THE THREE-DIMENSIONAL PROBLEM

It has been shown [2] that averaging the FDs of two different shapes (frequency domain) yields a FD which will inverse transform to a shape which appears to be an "average" contour, intermediate in shape between the two original contours. The data base which must be stored to represent a three-dimensional object can be reduced by using fewer projections, and "interpolating" between them in the frequency domain. This approach also enables a more accurate estimation to be made of the actual orientation in space of the object relative to the camera. Previous work on this estimation problem [3], [4] has assumed that the orientation of the unknown object is that of the nearest reference projection. This evidently limits estimation accuracy to the resolution of the reference projection set.

INTERPOLATION AND SAMPLING ERROR

One theoretical problem concerns the sample spacing used in sampling a time domain contour.

As explained in [5], a uniform sampling strategy is employed in the present algorithm, in order to facilitate analysis of a wide variety of shapes. While non-uniform sampling can result in faster convergence of a FD [2], there are obvious complications involved in defining such a sampling strategy for general shapes. When operations on FDs are made in the frequency domain, there is no guarantee that the resulting time domain representation will have uniform sampling. Hence, even if a contour appeared identical to an "average" contour computed by using linear combinations of known FDs, different sample spacing could result in some finite difference between the two FDs.

Consider the case in which an unknown projection lies directly between two library projections. Due to linearity, given two FDs a and b, a weighted sum of a and b transforms back as the same weighted sum of the transforms of a and b. It is easy to perform an experiment to measure the magnitude of this error. Simply computing the interpolated FD, inverse transforming, resampling uniformly, and transforming gives us two FDs whose M.S. distance is a measure of the point density error. Experiments with two shapes whose NFDs had a distance of about .3 (a distance less than .1 is generally used as a classification threshold) showed a point density error 190 to 200 times less than the distance between the original NFDs, with the interpolation coefficients equal to one half. The number of points used in the space domain vectors had a slight effect on the error, with more densely sampled vectors producing slightly less error. It can be concluded that this problem should not have a noticeable effect on the algorithm, since in practice, adjacent projections can be expected to have a NFD distance much less than .3. This fact should further minimize the point density error.

THE ESTIMATION PROBLEM

In the example discussed above, a projection was assumed to lie directly between two library projections, and the experiment was performed accordingly. In general, however, any random projection will not lie on the grid defined by a library of projections. Hence, more than two library projections must be used to perform the estimation. If a rectangular grid of projections is used, it would seem reasonable to do the estimation based on four library projctions, but consider instead the general case of estimating an N-vector X(k) as a linear combination of M N-vectors $Y_i(k)$, 1 < I < M:

$$\hat{X}(k) = \sum_{i=1}^{M} a_i \; Y_i(k) \qquad (2)$$

subject to the restriction that

$$\sum_{i=1}^{M} a_i = 1 \qquad (3)$$

It is straightforward to show that the optimum linear mean square estimate of X(k) is given by the solution to the equations:

$$\sum_{k=1}^{n} 2[X(k)-y_m(k)]D_i(k)$$

$$= \sum_{i=1}^{m-1} a_i(\sum_{k=1}^{n} D_i(k)D_j(k) + a_i D_i^2(k)) \quad 1 \leq i \leq m-1$$

with

$$D_i(k) \equiv Y_i(k)-Y_m(k) \text{ AND } a_m = 1 - \sum_{i=1}^{m-1} a_i$$

DATA REDUCTION

The time required to execute the above estimation algorithm is dependent on the dimension (N) of the vectors. It is thus desirable to reduce the dimension of the vectors as far as possible without degrading the classification performance. Equally important is the problem of storage of library data representing a three-dimension object. Previous FD classification results have indicated that there is no advantage in using more than 30 (complex) coefficients. In fact, quite good results have been obtained with only 14, although there was a slight degradation in performance when compared with 30.

The obvious approach is to estimate the autocorrelation matrix or covariance matrix of the data, and find the eigenvalues and eigenvectors which provide optimal data compression. There can be some difficulty in computing eigenvalues and eigenvectors of a 60 by 60 matrix. (Our feature vector consists of 30 complex coefficients.) One way to reduce the size of this matrix is to convert the data to 30 real coefficients. There are two ways that this might be done. The most obvious way is to simply take the magnitudes of the FD coefficients, since classifications based on magnitude information alone have been shown to be quite effective. However, if the data has bilateral symmetry, the associated NFDs should automatically be real. Even if the data does not have this symmetry, the normalization procedure tends to minimize the magnitudes of the imaginary parts of the NFD, and correspondingly minimize their contribution to the classification. Hence virtually all the information can be preserved by simply taking the real part of each NFD coefficient. This is the approach that was used in the experiment described below.

THE 3D ALGORITHM

An experiment was performed in which unknown aircraft outlines were identified and their orientation in space estimated using the above results. First, a set of aircraft was synthesized using a graphics approach. Three-dimensional approximations were constructed for six different aircraft, a mirage, a mig, a phantom, an F104, an F105, and a B57. Figure 5 shows representative images generated by this program. These three dimensional images were then rotated through appropriate angles to create a library of projections. The program was first given the library, and then given randomly selected orientations to identify.

The experiment of Dudani et al [3] was very similar, but several important differences should be noted. First, the data used by Dudani was constructed using model aircraft and a television camera hookup. It might appear that this is a more realistic approach, as well as a more demanding experiment than one using graphically generated data. However there are two problems with this method which the graphical method avoids. First, the resolution of the mechanical mount used by Dudani was 5 degrees. This represents an error in data generation which is avoided by the more exact graphics approach. Second, since the camera is a finite distance from the model, parallax problems affect the images, making the camera image different from the image received from a long distance. Since most practical photographs of actual aircraft in flight would be at a large distance, this error is undesirable.

In addition to the above considerations, it would probably be easier to use graphics techniques in a practical system, since accurate graphical representations constructed from blueprints would probably generate library data faster and more accurately than model-TV camera setups.

A major advantage of Dudani's approach is the accuracy of the aircraft shape. Our grahics program approximates each plane by using about 50-100 (geometric) planes. A more elaborate program could generate an arbitrarily accurate representation of each aircraft, with corresponding increase in computation time. The present data gives a reasonably good approximation to each aircraft, although the small detail is lacking. The effect on the classifications is probably to increase their difficulty, since certain minor features are missing. On the other hand, the data can probably be represented by fewer projections, due to the reduced complexity.

The basic problem considered by Dudani was classifying unknown aircraft images oriented at 5 degree intervals in a 140 degree by 90 degree sector. Each aircraft was represented by a library of moment feature vectors computed from 551 projections within this sector. The classification was performed by computing distances from a moment feature vector of the unknown image to the moment feature vectors of the library images, and then classifying using a distance-weighted k-nearest neighbor rule. Note that the images used by Dudani did not contain any mirror image pairs, and hence are obviously not all the images which can be theoretically recognized. In fact, a little reflection will convince one that if an object has enough assymetry, it can be recognized at any angle at all, and that angle identified. In the case of aircraft, there is generally bilateral symmetry, but this does not necessarily greatly limit the set of angles which can be theoretically recognized.

Our algorithm recognizes aircraft outlines taken from a sector of 180 by 180 degrees, i.e., a hemisphere. Note also that if the angles near the front view and rear view of the aircraft are deleted, the problem is much easier, since the shapes vary much more radically when large surfaces are viewed almost edgewise. Dudani's consideration of only 140 degrees reduces this problem. Our algorithm also recognizes random projections. There is no quantization of random projections corresponding to Dudani's 5 degree increments. Finally, the first version of our algorithm uses only 99 projections to represent an aircraft over the hemisphere, which represents a density of projections 14.3 times less than used by Dudani.

The actual classification program proceeds as follows. First, the library of projections is computed, and the NFD of each projection is computed. The autocorrelation matrix of the NFDs is computed, and an eigenvalue-eigenvector transformation reduces the data dimensionality from 30 complex numbers to 5 complex numbers. The real parts of the complex numbers are used to compute the autocorrelation matrix, but the complex parts of the transformed coefficients are kept to assist in the classification.

Next the M.S. distance from a given unknown contour to each library vector is computed. The distance to the nearest library contour is saved as the current best estimate of the minimum M.S. distance achievable. The projections adjacent to the nearest library projection are investigated by the M.S. estimation algorithm described above in an attempt to interpolate between the library projections.

The interpretation of the estimation coefficients returned by the estimation subroutine is somewhat heuristic, and goes something like this. This routine is constrained to return four numbers whose sum is 1.0. It often happens that two of the vectors being used to estimate the unknown contour are not very similar to the unknown contour, but are quite similar to each other. In this case, the estimation coefficients are of similar magnitudes and opposite sign, such as 2.0 and -2.05. What the estimation algorithm is doing is using the difference vector to help generate the optimum M.S. estimate of the unknown vector. We of course do not want to allow this kind of estimation, since it is inconsistent with our theory of interpolation of FDs. Another thing which is commonly observed when the unknown vector differs from the library vectors being used to estimate it is a set of large positive and negative estimation coefficients being returned. This again just tells us that we cannot expect to find a reasonable interpolated FD in the sector determined by that set of library projections.

The heuristic solution to these effects is as follows. First, we quit looking in a particular sector if the estimation coefficients returned are too large in magnitude. The algorithm is not very sensitive to this magnitude, and 1.5 to 2.0 is usually used. Also, if two coefficients sum to a small number (.1), but have relatively large magnitudes (>.5) they are assumed to be cancelling coefficients, and are deleted from the estimation set. The estimation process is then repeated with the remaining two vectors being used to estimate the unknown set. Similarly, any negative coefficients are deleted from an estimation, and the remaining two or three are used to repeat the estimation process. When an estimation of the unknown vector in terms of two, three, or four adjacent library projection vectors is achieved in which all the coefficients lie between zero and one, the distance is compared to the minimum distance achieved so far. If the new distance is less, the minimum distance is updated.

This process may be repeated for the k nearest library projections, where k is optional, and is generally in the range of 4-10. If the distances to the nearest k projections are approximately equal, the full k projections will be investigated. However, projections whose distances are more than 1.5 to 2.0 times greater than the minimum distance are not investigated. Each library projection has one, two, or four sectors surrounding it which must be investigated by the estimation subroutine. (If the sector is in the middle of the library set, there are four, if it is on the border, there are two, and if it is in a corner, there is one.) After the desired number of possible sectors are investigated, there are two possible procedures. The estimated orientation is taken to be that of the original nearest library vector, if the estimation fails to improve on this distance. If the estimation procedure is successful, the orientation is computed by multiplying the orientations of the vectors used in the estimation by their appropriate coefficients.

Results to data using 6 aircraft, and classifying 50 unknown images for each one show classification accuracy of 80% overall. The classification accuracy is about 72% if the estimation routine is not used.

It is expected that this figure can be improved by making use of the real and imaginary parts of the data when computing the eigenvector transformation matrix. Also, the spacing of library projections used in this experiment is non-uniform in an attempt to increase the Fourier space distance uniformity of the projections set, but it is not optimum. Finally, it may be necessary to increase the density of projections somewhat to bring the classification accuracy up to that achieved by Dudani. The current attempt to get by with a projection density at 14.3 times lower than Dudani may be too optimistic.

Given a chain code representation of an aircraft projection, the normalized FD is computed in about 2.76 sec. This time includes an FFT which is of length 512 or 1024. This normalized FD is then classified and its orientation estimated in about 2.38 sec. These times are for a PDP 11/45 with floating point hardware. The program itself is written in Fortran and is a research tool rather than a highly efficient implementation of the algorithm.

REFERENCES

1. G. H. Granlund, "Fourier Preprocessing for Hand Print Character Recognition," IEEE Trans. on Computers, Vol. C-21, pp. 195-201, Feb. 1972.

2. E. Persoon and K. S. Fu, "Sequential Decision Procedures with Prespecified Error Probabilities and Their Applications," School of Electrical Engineering, Purdue University, West Lafayette, IN, Tech. Rep. TR-EE 74-30, 1974.

3. S. A. Dudani et. al., "Aircraft Identification by Moment Invariants," IEEE Trans. on Computers, Vol. C-26, pp. 39-46, Jan. 1977.

4. C. W. Richard, Jr., and H. Hemami, "Identification of Three-Dimensional Objects Using Fourier Descriptors of the Boundary Curve," IEEE Trans. Syst., Man, and Cybern., Vol.. SMC-4, pp. 371-378, July 1974.

5. T. Wallace and P. A. Wintz, "Fourier Descriptors for Extraction of Shape Information," Final Report of Research for the Period Nov. 1, 1975-Oct. 31, 1976, ARPA Contract No. F30602-75-C-0150.

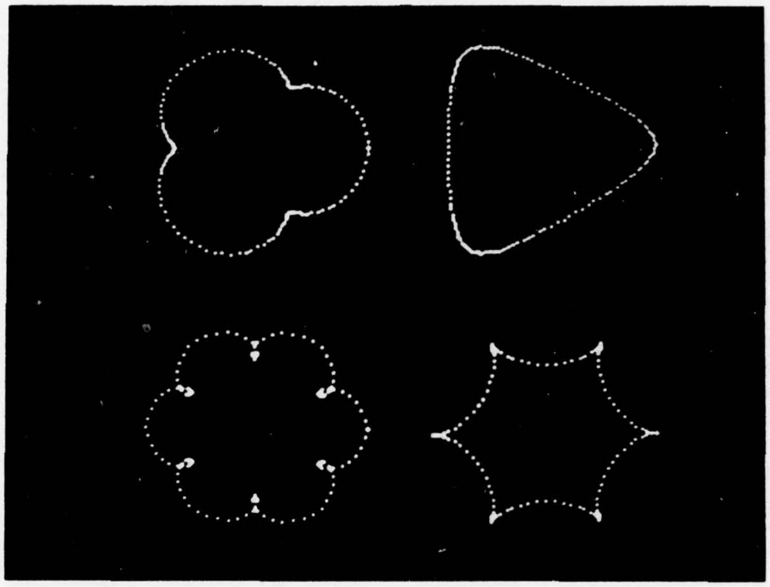


Fig. 1 Inverse Transforms of FD's Consisting of Selected Coefficients

Upper Left	
	A(1) = 1.0
	A(4) = 0.2
Upper Right	
	A(1) = 1.0
	A(-2) = 0.2
Lower Left	
	A(1) = 1.0
	A(7) = 0.2
Lower Right	
	A(1) = 1.0
	A(-5) = 0.2

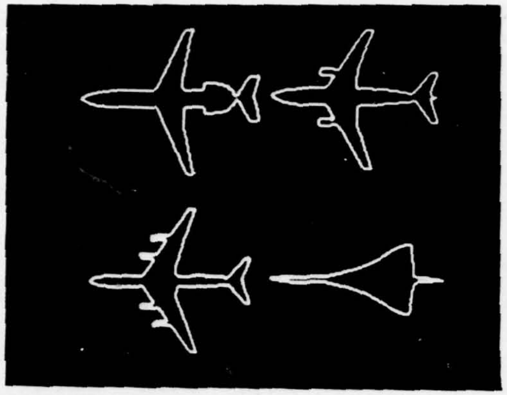


Fig. 2 Representative of High Res. Aircraft

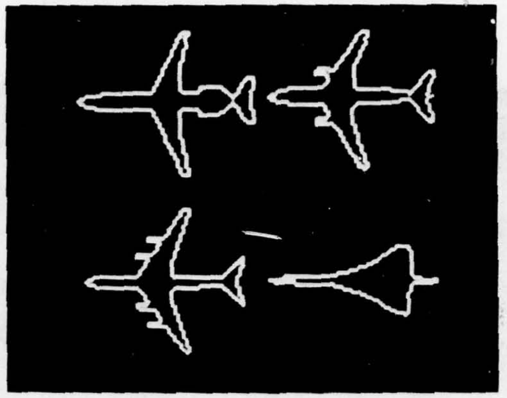


Fig. 3 Low Resolution Contours

BAC 111	AIRBUS A 300 B
DC 8 SERIES 50	BAC CONCORDE

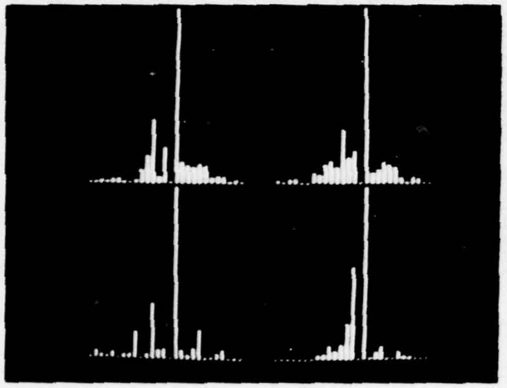


Fig. 4 Magnitudes of FD's Computed from Contours of Fig. 2

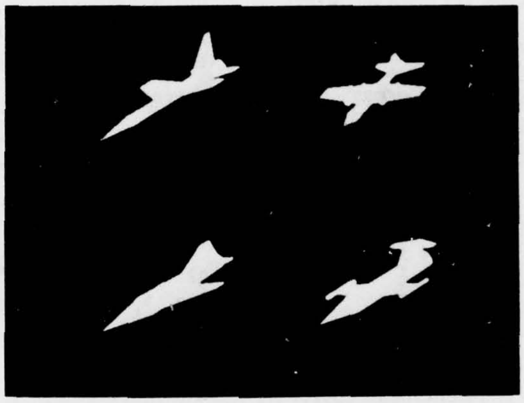


Figure 5 Representative Aircraft Images

Mirage	B 57
Phantom	F 104

IMAGE MENSURATION BY MAXIMUM A POSTERIORI PROBABILITY ESTIMATION

James W. Burnett and Thomas S. Huang
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

ABSTRACT

We present a fast algorithm for pulse width estimation from blurred and nonlinear observations in the presence of signal dependent noise. The main application is the accurate measurement of image sizes on film. The problem is approached by modeling the signal as a discrete position finite state Markov process, and then determining the transition location that maximizes the a posteriori probability. The method was applied to the measurement of the width of a road in an aerial photo taken at an altitude of 5000 feet. The resulting width estimate is accurate to within a few inches.

INTRODUCTION

This work presents a fast algorithm for pulse width estimation from blurred and nonlinear observations in the presence of signal dependent noise. The problem is motivated by the need for accurate measurements from remotely sensed photographs.

The problem is approached by modeling the signal (reflected light intensity) as a discrete position finite state Markov process. Sample functions of such a process are graphically represented by a path through a trellis. Blurred versions of these signals are similarly represented. By assigning a cost or length to each branch of the trellis a MAP sequence estimate of the signal is computed by finding the minimum cost or minimum length path through the trellis. MAP sequence estimates produced in this fashion have unambiguous edge locations making them useful for pulse width measurements.

The Viterbi algorithm is introduced as an efficient means of finding the minimum cost path through the trellis. When the possible states are known a-priori the algorithm produces asymptotically unbiased, minimum variance discrete width estimates. The decrease in performance is slight if the true states are unknown and estimates (obtained from the available data) used in their place.

Computer simulation results show the variance of the discrete estimates is close to the Cramer-Rao bound. The algorithm is applied to the measurement of a road in an aerial photo taken at an altitude of 5000 feet. The resulting width estimate is accurate to within a few inches. Experimental results also indicate the estimates are not sensitive to small variations in the degrading system.

MODEL FOR A STEP EDGE

A scan line across a step edge is characterized by an initial level a_1 at position 1, a final level a_2 at position M and an abrupt transition between levels a_1 and a_2 somewhere between positions 1 and M. If the $k^{\underline{th}}$ sample along the line has value a_1 the next sample i_{k+1} can assume the value a_1 with some probability (say p_{11}) or can assume the value a_2 with some probability $p_{12} = 1-p_{11}$. If the sample i_k had a value of a_2 then i_{k+1} must also be a_2. Thus a simple Markov model for step edges is:

$$Pr(i_1=a_1) = 1$$

$$Pr(i_M=a_2) = 1$$

$$Pr(i_{k+1}=a_j|i_1,\ldots,i_k) = Pr(i_{k+1}=a_j|i_k)$$

where

$$Pr(i_{k+1}=a_1|i_k=a_1) = p_{11}$$

$$Pr(i_{k+1}=a_2|i_k=a_1) = p_{12}$$

$$P_r(i_{k+1}=a_2|i_k=a_2) = 1 \qquad (1)$$

This model can be represented graphically as shown in Figure 1(a). The nodes represent possible intensity levels at each position k. The dotted lines between nodes (branches) represent possible transitions between levels. Each branch has been labeled with the transition probability $P(i_{k+1}|i_k)$ that corresponds to the nodes the branch connects. Each possible path through the trellis along the dotted lines represents a different edge location. Figure 1(b) models the case where there is uncertainty about the levels at the initial or final position. In particular paths from position 1 to position M along the top or bottom of the trellis correspond to the absence of an edge.

MODEL FOR A BLURRED EDGE

Assume that a real or blurred edge is adequately modeled by the output of a blurring system h when the input to h is the ideal edge. The system h may be nonlinear; however, it is assumed that the $k^{\underline{th}}$ sample of the output of h, y_k, depends only on the 2v+1 adjacent inputs $(i_{k-v},\ldots,i_k,\ldots,i_{k+v}) \triangleq \xi_k$ for some $v < \infty$ and that there is a one to one correspondence between the output sequence $\underline{Y} = (y_1,\ldots,y_M)$ and the input state sequence $\underline{\xi} = (\xi_1,\ldots,\xi_M)$. Since there are only a finite number of values that i_k can assume (two in the case of step edges) there are only a finite number of states ξ_k. *We denote the input* sample sequence by $I = (i_1,\ldots,i_M)$.

From (1):

$$Pr(y_1 = h(a_1,\ldots,a_1)) = 1$$

$$Pr(y_{k+1} = h(a_1,\ldots,a_1)|y_k=h(a_1,\ldots,a_1)) = p_{11}$$

$$Pr(y_{k+1} = h(a_1,\ldots,a_1,a_2)|y_k=h(a_1,\ldots,a_1)) = p_{12}$$

$$Pr(y_{k+1} = h(a_1,\ldots,a_2,a_2)|y_k=h(a_1,\ldots,a_1,a_2)) = 1$$

$$\vdots$$

$$Pr(y_{k+1} = h(a_2,\ldots,a_2)|y_k=h(a_1,a_2,\ldots,a_2)) = 1$$

$$Pr(y_M = h(a_2,\ldots,a_2)) = 1 \qquad (2)$$

Thus a blurred edge can be represented by a path through a trellis as shown in Figure 2.

Since there is a one to one correspondence between $\underline{Y}$, $\underline{\xi}$, and $\underline{I}$ any one of the three can be uniquely represented by a path through the trellis.

MAXIMUM A POSTERIORI PROBABILITY SEQUENCE ESTIMATION

The maximum a-posteriori probability (MAP) estimate of a sequence $\underline{I}$ given a sequence $\underline{Z}$($\underline{Z}$ being a degraded and noisy version of I, is defined as a sequence $\hat{\underline{I}} = (\hat{i}_1,\hat{i}_2,\ldots,\hat{i}_M)$ such that $P(\underline{I}|\underline{Z})_{I} = \hat{I}$ is a maximum. To calculate $\hat{I}$ a model is needed for the relationship betwen $\underline{I}$ and $\underline{Z}$. The assumed observation model is shown in Figure 4. $\underline{I} = (i_1,i_2,\ldots,i_M)$ is the sequence of ideal light intensities with i_k the light intensity of the k^{th} sample point entering the imaging system. Each i_k *can assume one of* G possible values $a_1,\ldots,a_G$. For example $\underline{I}$ might represent the sequence of reflected light intensities from a scan line of an aerial photo of a bridge across a river. In this case there would be two possible intensity levels: a_1 corresponding to the light reflected from the water and a_2 corresponding to the light *reflected* from concrete (or whatever construction material was used in the bridge). The state at position k, ξ_k, is defined to be a set of adjacent intensities $(i_{k-v},\ldots,i_k,\ldots,i_{k+v})$. Since each i_j can assume only a *finite number* of values each ξ_k is one of a finite set $[S_1 \ldots S_q]$. Further (to within boundary conditions) there is a one to one correspondence between the state sequence $\underline{\xi}$ and the intensity sequence $\underline{I}$.

The system h(•) represents the degradation of the sequence $\underline{I}$. In the case of photographic imagery this includes blurring due to lense defects, scattering, diffraction, camera motion, etc. as well as the nonlinear relationship between light intensity and film density. The only assumption on h is that there is a *one to one correspondence* between $\underline{Y} = (y_1,\ldots,y_M)$ (where $y_k = h(\xi_k)$) and $\underline{\xi}$.

$\underline{N}$ is a sequence of independent noise samples. The parameters of the noise distribution may *depend on the signal. For example* film grain noise is approximately normal with a standard deviation approximately proportional to a power of the signal level.

By definition the MAP sequence estimate $\hat{\underline{I}}$ of $\underline{I}$ is

$$P(\underline{I}|\underline{Z})_{I=\hat{I}} \text{ is a maximum} \qquad (3)$$

but since there is a one to one correspondence between $\underline{I}$ and $\underline{\xi}$, (3) is equivalent to

$$P(\underline{\xi}|\underline{Z})_{\underline{\xi}=\hat{\underline{\xi}}} \text{ is a maximum} \qquad (4)$$

However, by Bayes rule, the independence of the noise and a Markov assumption on $\underline{\xi}$ (4) is equivalent to maximizing:

$$\prod_{\ell=1}^{M} P(\xi_{\ell+1}|\xi_\ell)\; p(z_\ell|h(\xi_\ell)) \qquad (5)$$

or equivalently minimize

$$\sum_{\ell=1}^{M} -\log P(\xi_{\ell+1}|\xi_\ell)-\log p(z_\ell|h(\xi_\ell)) \triangleq \sum_{\ell=1}^{M} \Gamma(\xi_\ell) \quad (6)$$

By assigning a cost or length of $\Gamma(\xi_\ell)$ to each branch of a trellis it is easy to see that the MAP estimate $\hat{\xi}$ represents the lowest cost or minimum length path through the trellis.

There is a very good algorithm for finding the minimum cost path through a trellis due to Viterbi [1,2] called the Viterbi algorithm (VA).

UNKNOWN LEVELS

The previous sections assumed the levels $a_1,a_2,\ldots$ that the scan line could assume were known *a-priori. In practice this may not be true.* In this case a reasonable course of action is to obtain "training" samples of levels characterizing the object to be measured and its background, and then estimate the level values from these training samples.

MEASUREMENT OF A ROAD

A 1:5000 scale black and white negative taken with Kodak Plus X Aerographic film from an altitude of 5000 feet was obtained and digitized on a flying spot scanner. The scene (sampled at a rate of 24 samples/mm) is shown in Figure 4 and contains an intersection of two gravel roads in Warren County, Indiana. Figure 5 shows one of the roads (sampled at a rate of 96 samples/mm) and

Figure 6 shows a scan line across the road of Figure 5. Five hundred training samples from one of the roads showed the average density was .942 with a variance of .00213. One thousand training samples from the field surrounding the road had an average density of .669 with a variance of .00236. The nominal film properties were obtained from Tarkington [3] and Paris [4]. The frequency response of the image blur was assumed to be the product of the film frequency response

$$T_1(f) = \frac{1}{1+(\frac{2\pi f}{250})^2} \qquad (7)$$

and the response of an ideal diffraction limited lens with a cutoff frequency of half the sampling rate:

$$T_2(f) = \frac{2}{\pi}\left(\cos^{-1}\left(\frac{f}{f_c}\right) - \frac{f}{f_c}\sqrt{1-\left(\frac{f}{f_c}\right)^2}\right)$$

$$f_c = 48 \text{ cycles/mm} \qquad (8)$$

Figure 7 shows the line spread function corresponding to equations (7) and (8).

Ten independent measurements of one of the roads were made and the results shown in Table 1. The variance of each y_k was taken to be .00213 if y_k corresponded to a state where i_k was a sample from the road or .00236 if y_k corresponded to an i_k from the field.

The variance of the digital measurement was 1.15 sample points squared and the uncertainty indicated in Table 1 represents plus or minus two standard deviations.

Ten optical measurements were made with a 50 power magnifier and a reticle marked in thousandths of an inch. Table 1 shows the results and uncertainty of this measurement. Again the uncertainty represents two standard deviations.

The site of the road was visited and the width found to be 18' 11" with a tape measure. There is a fair amount of uncertainty connected with this measurement. The edges of the road are characterized by vegetation which can overhang or encroach upon the road by several inches on either side. Measurements on similar roads varied from 18' 6" to 19' 10". Therefore, the true width of the road the day the photograph was taken is not known exactly.

Table 1. Road Width Measurement Results

Method	Road Width on Film	Width on Ground
VA	110.2 s.p. ± 2.1	5.74 m ± .11 (18'10" ± 4")
Optical	.0456" ± .0024	5.79 m ± .30 (19'2" ± 12")
Tape Measure	-----	5.76 m ± .15 (18'11" ± 6")

Finally, the effect of the cutoff frequency f_c in equation (8) was examined. Line spread functions for different values of f_c were calculated and the ten measurements were repeated. The results are shown in Table 2.

Table 2. The Effect of f_c on Width Estimates

f_c (cycles/mm)	Width (sample points)	Variance
56	110.3	1.20
48	110.2	1.15
40	109.8	1.1
32	109.1	.96

Table 2 indicates the width estimates produced by the VA are not overly sensitive to imperfect knowledge of the degrading system.

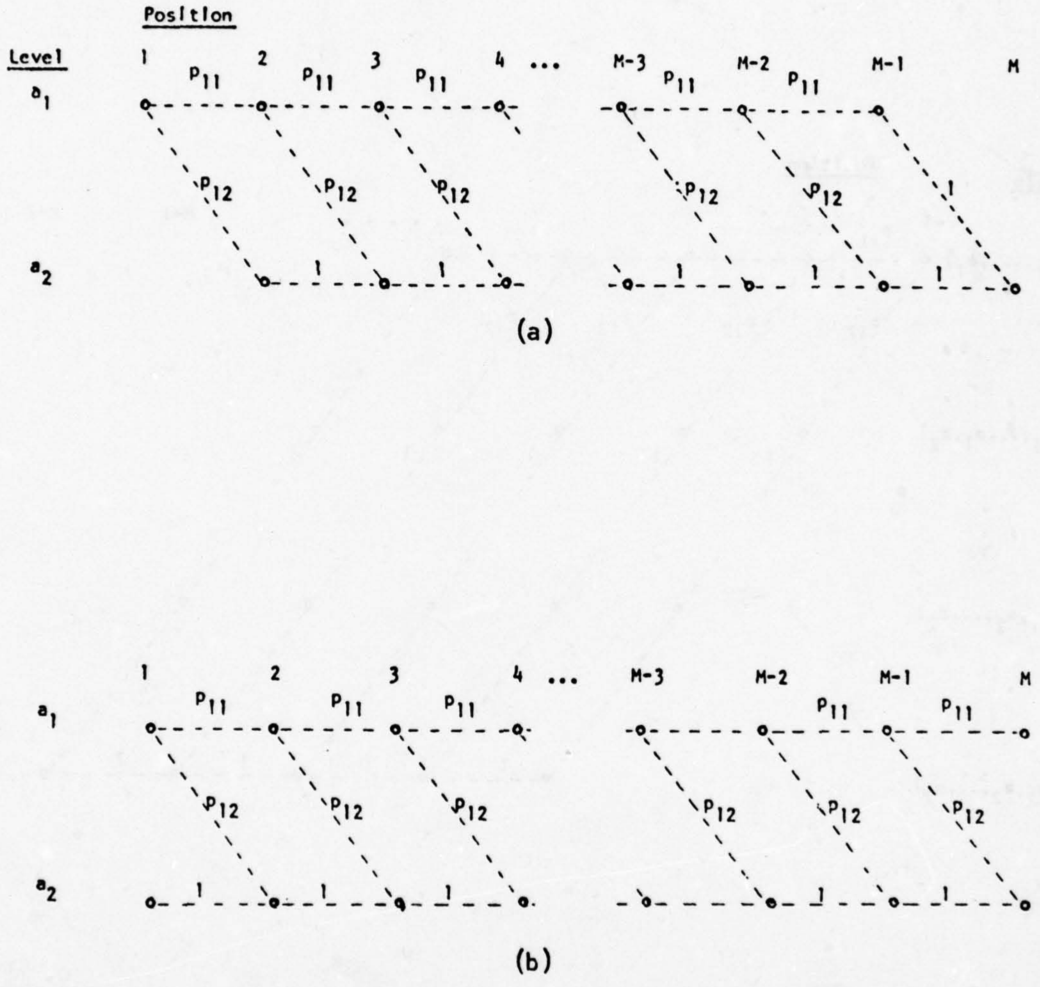
REFERENCES

1. A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," IEEE Trans. Inf. Theory, vol. IT-13, April 1967.

2. G. Forney, "The Viterbi algorithm," Proc. IEEE, vol. 61, March 1973.

3. R. Tarkington, "Kodak panchromatic negative films for aerial photography," Photogrammetric Engineering, December 1959, pp. 695-699.

4. D. Paris, "Approximation of the sine wave response of photographic emulsions," JOSA, vol. 51, Sept. 1961, pp. 988-991.



(a)

(b)

Fig. 1 Trellises for a step edge.

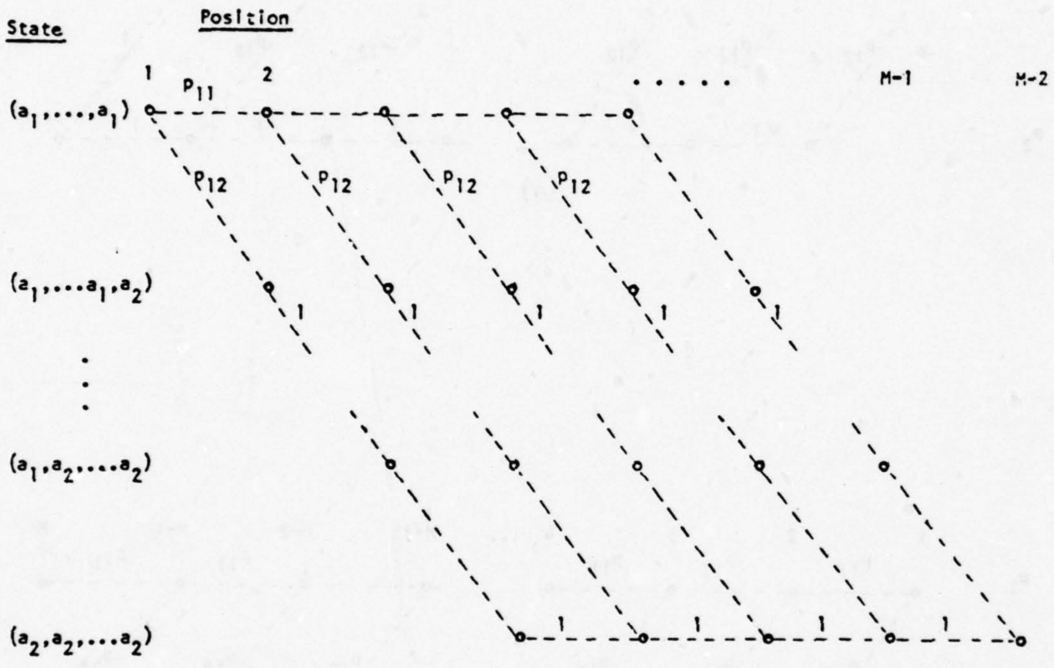


Fig. 2 Trellis for a blurred step edge.

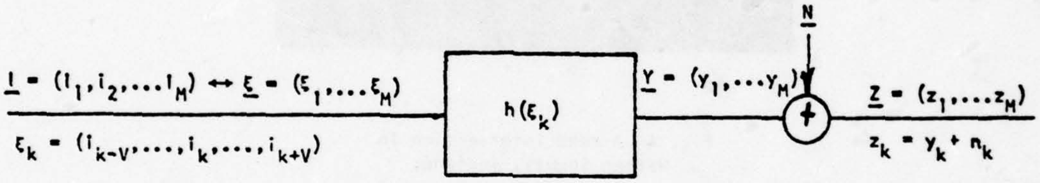

$\underline{I} = (i_1, i_2, \ldots. i_M) \leftrightarrow \underline{\xi} = (\xi_1, \ldots. \xi_M)$
$\xi_k = (i_{k-V}, \ldots., i_k, \ldots., i_{k+V})$
$h(\xi_k)$
$\underline{Y} = (y_1, \ldots y_M)$
$\underline{N}$
$+$
$\underline{Z} = (z_1, \ldots. z_M)$
$z_k = y_k + n_k$

Fig. 3 The observation model

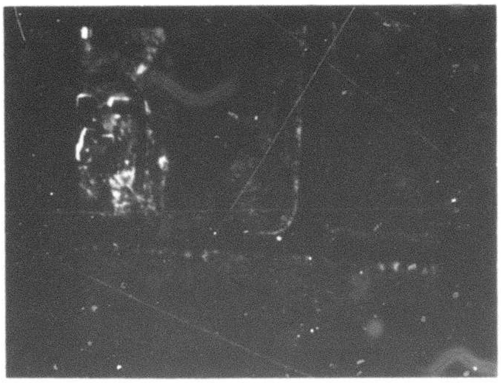


Fig. 4 A road intersection in Warren County, Indiana.

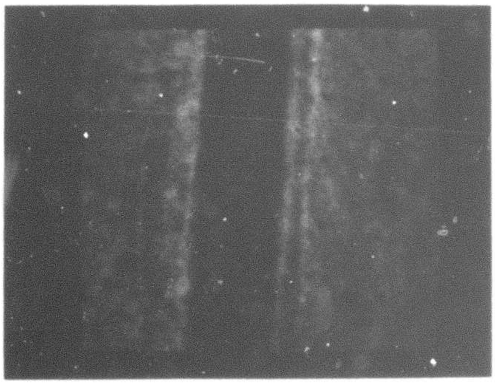


Fig. 5 Enlargement of a section of one of the roads in Fig. 4.

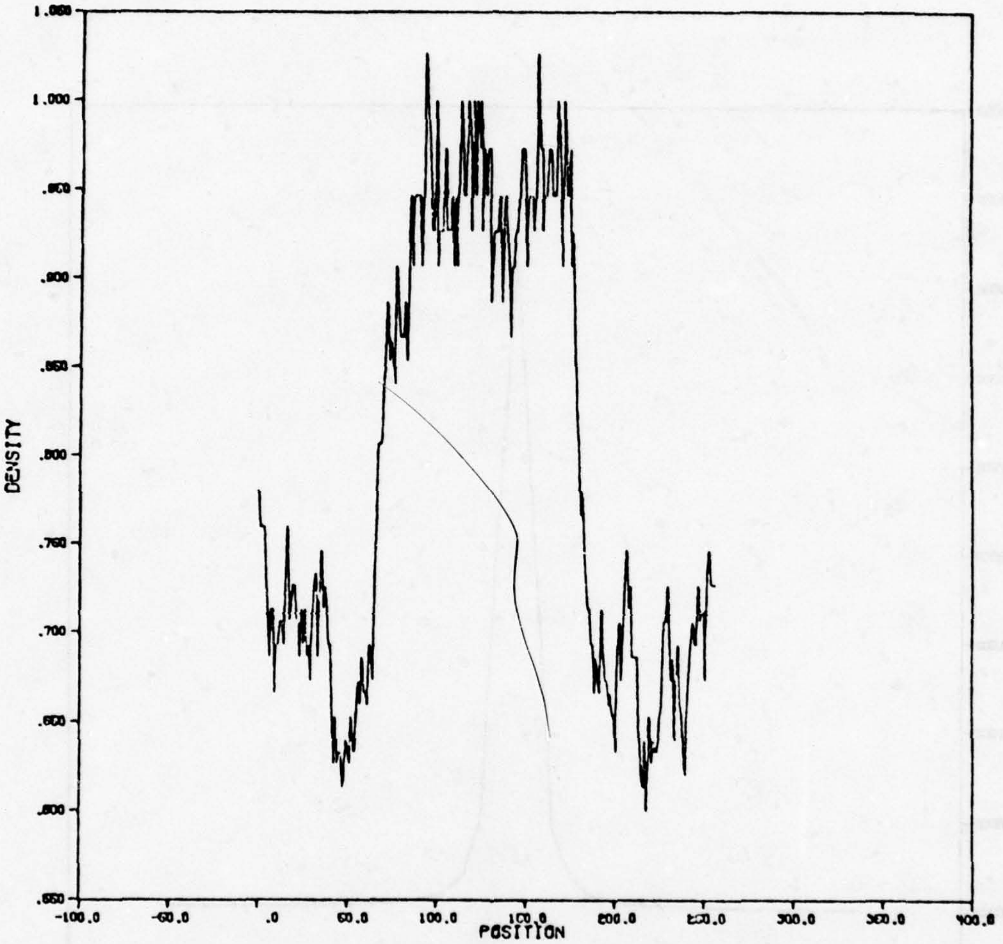


Fig. 6 A scan line across the road in Fig. 5.

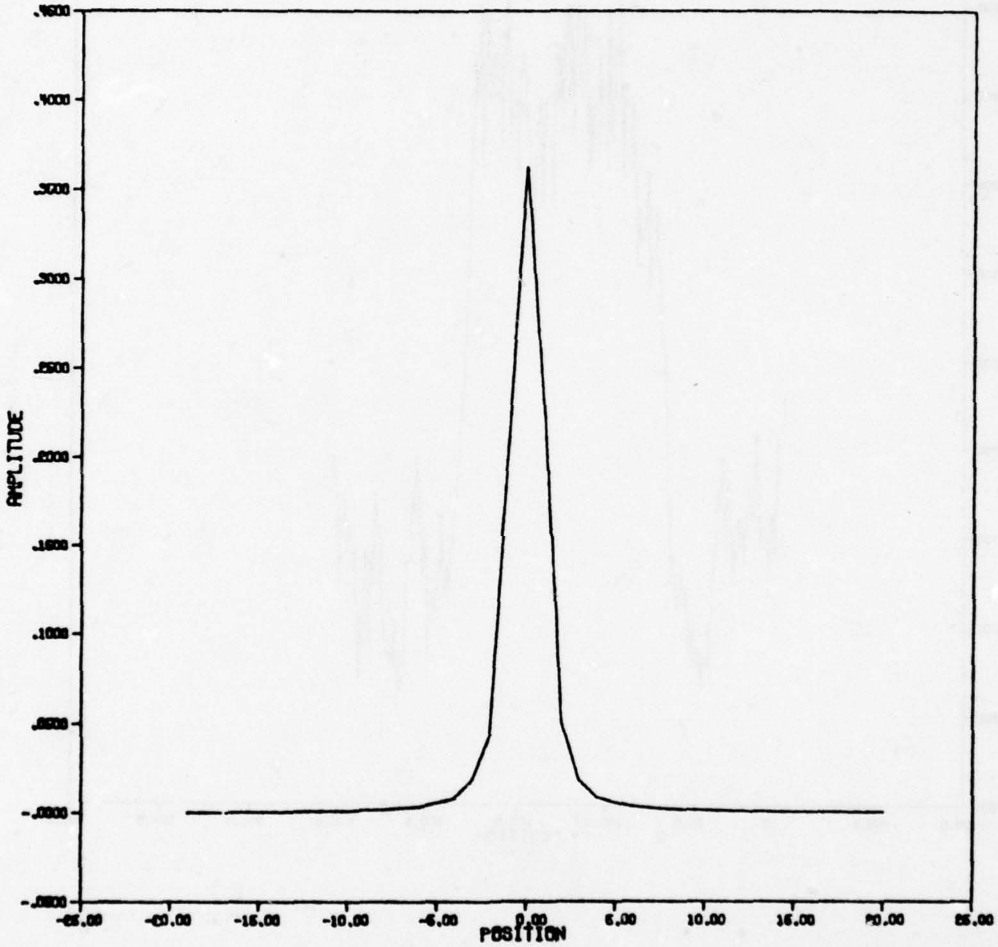


Fig. 7 Line spread function of the film-lens combination.

ADAPTIVE THRESHOLD FOR AN IMAGE RECOGNITION SYSTEM

D. Serreyn and R. Larson

HONEYWELL INC.
Systems and Research
Minneapolis, Minnesota 55413

ABSTRACT

An adaptive object extraction algorithm (Autothreshold), using pixel classification and Sobel edges, has been implementated in hardware and tested on recorded real time FLIR imagery. The results show improvement in man-made object detection and reveal certain problems with using commercially available CCD's.

AUTOSCREENER

A major problem in real time target image recognition is the large bandwidth of the data. The information bandwidth must be reduced by orders of magnitude before recognition can be performed. The Autoscreener performs this bandwidth reduction in two stages. The first is image segmentation, that extracts subimages of potential interest. The second is to classify these as either man made objects (MMO) or clutter. The MMO's are then the low bandwidth input to the recognition algorithm.

In FLIR imagery, hot areas are generally associated with targets. However, intensity thresholding produces poor image segmentation. Edge information improves the segmentation. The Autoscreener extracts those parts of the image where the intensity exceeds the background estimate and which are bounded by edges. The AFAL funded portion of this contract develops a means of adapting the thresholds on edge and intensity to producε segments that are insensitive to changes in scene contrast and average intensity.

EDGE EXTRACTION

Edge detection is done by using a 3 x 3 Sobel operator with large values giving an indication of a possible object of interest. The Sobel edge is implemented using commercially available CCD's for line delays and for pixel delays. Clock noise and interscan line voltage levels are major problems that were overcome. Circuit layout is also very critical in using these devices. On the positive side, these devices are usable for scan line delays of three milliseconds without noticeable signal degradation.

The absolute value of the horizontal edge component is thresholded to generate a logical edge signal. The vertical component was not used because of banding in the FLIR data. (The banding is due to improper balancing of the detector at the time the data was taken. It is generally not noticeable while observing several sequential frames but is noticeable when a single frame is used.)

The edge threshold adapting is based upon the scan line averages. The edge information is averaged over one line of the image. At the end of the scan line the average is sampled and held for the next scan line. This sampled value is then multiplied by a constant to provide the threshold for the incoming edge information.

BACKGROUND ESTIMATOR

The Autothreshold makes the Autoscreener self adaptive to background and contrast changes. The intensity adapting is done by estimating the background intensity at each pixel location while the picture is being scanned.

Two background estimation methods have been tested. One method is controlled by the pixel classifier. This classifier uses intensity changes to detect "target like" portions of the image and the image and the background estimate is updated only at the "non-target like" pixels. The other method simply uses a two dimensional lowpass filter to estimate the background.

The bright information is obtained by subtracting the background estimate from the video intensity. The difference is thresholded using the variability of adjacent background scan line estimates.

CONCLUSIONS

In addition to the hardware results, the following observations may be of value to other investigators:

- Line by line adaption of intensity and edge thresholds is superior to determining threshold values from frame averages.

- Pixel classification, based on intensity differences, produces an image segmentation of similar quality to that produced by the combined edge and bright signals.

- Detector equalization is critical in using the Sobel algorithm on FLIR imagery.

The MMO classifier is being evaluated at this time and these results will be reported at the workshop.

SESSION IV

REGISTRATION & SEGMENTATION

USING SYNTHETIC IMAGES TO REGISTER REAL IMAGES WITH SURFACE MODELS

Berthold K. P. Horn and Brett L. Bachman

Artificial Intelligence Laboratory, Massachusetts Institute of Technology
545 Technology Square, Cambridge, MA 02139 U.S.A.

ABSTRACT

A number of image analysis tasks can benefit from registration of the image with a model of the surface being imaged. Automatic navigation using visible light or radar images requires exact alignment of such images with digital terrain models. In addition, automatic classification of terrain, using satellite imagery, requires such alignment to deal correctly with the effects of varying sun angle and surface slope. Even inspection techniques for certain industrial parts may be improved by this means.

We achieve the required alignment by matching the real image with the synthetic image obtained from a surface model and known positions of the light sources. The synthetic image intensity is calculated using the reflectance map, a convenient way of describing the surface reflection as a function of surface gradient. We illustrate the technique using LANDSAT images and digital terrain models.

MOTIVATION

Interesting and useful new image analysis methods may be developed if registered image intensity and surface slope information is available. Automatic change detection, for example, seems unattainable without an ability to deal with variations of appearance with changes in the sun's position. In turn, these variations can be understood only in terms of surface topography and reflectance models. Similarly, human cartographers consult both aerial photographs and topographic maps of a region to establish the location of streamlines. Automatic analysis of either of these information sources alone is unlikely to lead to robust methods for performing this task.

Accurate alignment of images with surface models is therefore an important prerequisite for many image understanding tasks. We describe here an automatic method of potentially high accuracy that does not depend on feature extraction or other sophisticated image analysis methods. Instead, all that is required is careful matching of the real with a synthetic image. Because this is an area-based process, it has the potential for sub-pixel accuracy -- accuracy not attainable with techniques dependent on alignment of linear features such as edges or curves. The method is illustrated by registering LANDSAT images with digital terrain models.

POSSIBLE APPROACHES

One way to align a real image with a surface model might be through the use of a reference image obtained under controlled conditions. New images could then be matched against the reference image to achieve alignment. Unfortunately, the appearance of a surface depends quite dramatically on the position of the light source (as seen in figure 1, for example), so that this method works only for a limited daily interval for a limited number of days each year [1]. This problem disappears when one uses synthetic images, since the position of the source can be taken into account.

A more sophisticated process would not match images directly, but first perform a feature extraction process on the real image and then match these features with those found in the reference image. One finds, however, that different features will be seen when lighting changes: for example, ridges and valleys parallel to the illumination direction tend to disappear (see figure 1 again). In addition, the apparent position of a feature as well as its shape may depend somewhat on illumination. More serious may be the present feature extraction schemes' computational cost and lack of robustness. Finally, we should note that the accuracy obtained by matching linear features is likely to be lower than that obtainable with a method based on an aerial match.

One might consider calculating the shape of the surface from intensities in the image [2]. This, however, is computationally expensive and not likely to be very accurate in view of the variation in the nature of surface cover. A more accurate method, estimating the local gradient using similar methods [3] and then matching these with gradients stored in the terrain model, still involves a great deal of computation.

The method chosen here depends instead on matching the real image with a synthetic image produced from the terrain model. The similarity of the two images depends in part upon how closely the assumed reflectance matches the real one. For mountainous terrain and for images taken with low sun elevations, rather simple assumptions about the reflectance properties of the surface gave very good results. Since all LANDSAT images are taken at about 9:30 local solar time, the sun elevations in this case are fairly small and image registration for all but flat terrain is straightforward.

This implies that LANDSAT images are actually not optimal for automatic terrain classification, since the intensity fluctuations due to varying surface gradients often swamp the intensity fluctuations due to variations in surface cover. An important application of our technique in fact is the removal of the intensity fluctuations due to variations in surface gradient from satellite images in order to facilitate the automatic classification of terrain. To do this, we must model the way the surface reflects light.

THE REFLECTANCE MAP

Work on image understanding has led to a need to model the image-formation process. One aspect of this concerns the geometry of projection, that is, the relationship between the position of a point and the coordinates of its image. Less well understood is the problem of determining image intensities, which requires modelling of the way surfaces reflect light. For a particular kind of surface and a particular placement of light sources, surface reflectance can be plotted as a function of surface gradient (magnitude and direction of slope). The result is called a reflectance map and is usually presented as a contour map of constant reflectance in gradient space [3].

One use of the reflectance map is in the determination of surface shape from intensities [2] in a single image; here, however, it will be employed only in order to generate synthetic images from digital terrain models.

DIGITAL TERRAIN MODELS

Work on computer-based methods for cartography, prediction of side-looking radar imagery for flight-simulators, automatic hill-shading and machines that analyze stereo aerial photography has led to the development of digital terrain models. These models are usually in the form of an array of terrain elevations, z_{ij}, on a square grid.

Data used for this paper's illustrations was entered into a computer after manual interpolation from a contour map and has been used previously in work on automatic hill-shading [4,5]. It consists of an array of 175-240 elevations on a 100-meter grid corresponding to a 17.5 km by 24 km region of Switzerland lying between 7°1' East to 7°15' East and 46°8.5' North to 46°21.5' North. The vertical quantization is 10 meters, and elevations range from 410 meters (in the Rhone valley) to 3210 meters (on the Sommet des Diablerets). The topographic maps used in the generation of the data are "Les Diablerets" (No. 1285) and "Dent de Morcles" (No. 1305), both on a 1:25 000 scale [6]. Extensive data editing was necessary to remove entry errors; some minor distortions of elevations may have resulted.

Manually-entered models of two regions in Canada have also been used [5,7]. Another set, covering a region of California, was produced by a digital simulator of a proposed automatic stereo scanner. (Output of two experimental automatic stereo scanners, one built at ETL [8] and one built at RADC [9], could not be obtained).

The United States Geological Survey [10] supplies digital terrain models on magnetic tape, each covering one square degree of the United States, with a grid spacing of about 208 feet (63.5 m). These models apparently were produced by interpolation from hand-traced contours on existing topographic maps of the 1:250 000 series. Interpolation to a resolution of .01 inch (0.254 mm) on the original maps fills in elevations between the contours spaced 200 feet (60.96 m) vertically. The final result is smoothed and "generalized" to a considerable extent; nevertheless, this is the most prolific source of surface models available to the public.

THE GRADIENT

A gradient has two components, namely the surface slope along two mutually perpendicular directions. If the surface height, z, is expressed as a function of two coordinates x and y, we define the two components, p and q, of the gradient as the partial derivatives of z with respect to x and y respectively. In particular, a Cartesian coordinate system is erected with the x-axis pointing east, the y-axis north and the z-axis up. Then, p is the slope of the surface in the west-to-east direction, while q is the slope in the south-to-north direction:

$$p = \frac{\partial z}{\partial x} \qquad q = \frac{\partial z}{\partial y}$$

One can estimate the gradient from the digital terrain model using first differences,

$$p = [z_{(i+1)j} - z_{ij}]/\Delta$$

$$q = [z_{i(j+1)} - z_{ij}]/\Delta$$

where Δ is the grid-spacing. More sophisticated schemes are possible [5] for estimating the surface gradient, but are unnecessary.

POSITION OF THE LIGHT SOURCES

In order to be able to calculate the reflectance map, it is necessary to know the location of the light source. In our case the primary source is the sun, and its location can be determined easily by using tables intended for celestial navigation [11, 12, 13] or by straightforward computations [14, 15, 16, 17]. In either case, given the data and time, the azimuth (θ) and the elevation (ϕ) of the sun can be found. Here, azimuth is measured clockwise from North, while elevation is simply the angle between the sun and horizon (see figure 2). Now one can erect a unit vector at the origin of the coordinate system pointing at the light source,

$$\hat{\underline{n}}_S = [\sin(\theta)\cos(\phi), \cos(\theta)\cos(\phi), \sin(\phi)].$$

Since a surface element with gradient (p,q) has a normal vector $\underline{n} = (-p, -q, 1)$, we can identify a particular surface element that happens to be perpendicular to the direction towards the light source. Such a surface element will have a surface normal $\underline{n}_S = (-p_S, -q_S, 1)$ where $p_S = \sin(\theta)\cot(\phi)$ and $q_S = \cos(\theta)\cot(\phi)$. We can use the gradient (p_S, q_S) as an alternate means of specifying the position of the source.

In work on automatic hill-shading, for example, one uses $p_S = -0.707$ and $q_S = 0.707$ to agree with standard cartographic conventions which require that the light source be in the North-west at 45° elevation ($\theta = 7/4$) , $\phi = \pi/4$ [5].

REFLECTANCE AS A FUNCTION OF THE GRADIENT

Reflectance of a surface can be expressed as a function of the incident angle (i), the emittance angle (e) and the phase angle (g) (see figure 3). We use a simple, idealized reflectance model for the surface material,

$$\Phi_1(i, e, g) = \rho \cos(i)$$

This reflectance function models a surface which, as a perfect diffuser, appears equally bright from all viewing directions. Here, ρ is an "albedo" factor and the cosine of the incident angle simply accounts for the foreshortening of the surface element as seen from the source. More sophisticated models of surface reflectance are possible [3], but are unnecessary for this application.

The incident angle is the angle between the local normal (-p, -q, 1) and the direction to the light source $(-p_S, -q_S, 1)$. The cosine of this angle can then be found by taking the dot-product of the corresponding unit vectors,

$$\cos(i) = \frac{(1 + p_S p + q_S q)}{\sqrt{1 + p_S^2 + q_S^2}\ \sqrt{1 + p^2 + q^2}}$$

Finally,

$$\Phi_1(p,q) = \frac{\rho(1 + p_S p + q_S q)}{\sqrt{1 + p_S^2 + q_S^2}\ \sqrt{1 + p^2 + q^2}}$$

Another reflectance function, similar to that of materials in the maria of the moon and rocky planets [2, 18], is a little easier to calculate:

$$\Phi_2(p,q) = \rho \cos(i)/\cos(e) = \frac{\rho(1 + p_S p + q_S q)}{\sqrt{1 + p_S^2 + q_S^2}}$$

This reflectance function models a surface which reflects equal amounts of light in all directions. For small slopes and low sun elevations, it is very much like the first one, since then $(1 + p^2 + q^2)$ will be near unity. Both functions were tried and both produce good alignment -- in fact, it is difficult to distinguish synthetic images produced using these two reflectance functions.

SYNTHETIC IMAGES

Given the projection equations that relate points on the objects to images of said points, and given a terrain model allowing calculation of surface gradient, it is possible to predict how an image would look under given illuminating conditions, provided the reflectance map is available. We assume simple orthographic projection here as appropriate for a distant spacecraft looking vertically down with a narrow angle of view. Perspective projection would require a few minor changes in the algorithm.

The process of producing the synthetic image is simple. An estimate of the gradient is made for each point in the digital terrain model by considering neighboring elevations. The gradient's components, p and q, are then used to look up or calculate the expected reflectance. An appropriate intensity is placed in the image at the point determined by the projection equation. All computations are simple and local, and the work grows linearly with the number of picture cells in the synthetic image.

Sample synthetic images are shown in figure 1. The two images are of the same region with differences in assumed location of the light source. In figure 2a the sun is at an elevation of 34° and azimuth of 153°, corresponding to its true position at 9:52 G.M.T., 1972/Oct/9, while for figure 2b it was at an elevation of 28° and an azimuth of 223°, corresponding to its position at 13:48 G.M.T. later on the same day. The corresponding reflectance maps are shown in figure 4.

Reflectance maps for the simpler reflectance function $\Phi_2(p,q)$ under the same circumstances are shown in figure 5. Note that near the origin there is very little difference between $\Phi_1(p,q)$ and $\Phi_2(p,q)$. Since most surface elements in this terrain model have slopes less than $1/\sqrt{2}$, as shown in the scattergram (see figure 6), synthetic images produced using these two reflectance maps are similar.

Since the elevation data is typically rather coarsely quantized as a result of the fixed contour intervals on the base map, p and q usually take on only a few discrete values. In this case, it is convenient to establish a lookup table for the reflectance map by simply precalculating the reflectance for these values. Models with arbitrarily complex reflectance functions can then be easily accomodated as can reflectance functions determined experimentally and known only for a discrete set of surface orientations.

Since the real image was somewhat smoothed in the process of being reproduced and digitized, we found it advantageous to perform a similar smoothing operation of the synthetic images so that the resolution of the two approximately matched. Alignment of real and synthetic images was, however, not dependent on this refinement.

THE REAL IMAGE

The image used for this paper's illustrations is a portion of a LANDSAT [1, 19] image acquired about 9:52 G.M.T. 1972/Oct/9 (ERTS-1 1078-09555). We used channel 6 (near infra-red, .7μ to .8μ), although all four channels (4, 5, 6, & 7) appeared suitable -- with channel 4 (green, .5μ to .6μ) being most sensitive to water in the air column between the satellite and the ground, and channel 7 best at penetrating even thin layers of clouds and snow. Figure 7 compares an enlargement of the original transparency with the synthetic image generated from the terrain model.

A slow-scan vidicon camera (Spatial Data Systems 108) was used to digitize the positive transparency of 1:1 000 000 scale. Individual picture cells were about .1 mm on a side in order to match roughly the resolution of the synthetic image data. In recent work, we used a more accurate version digitized on a drum-scanner (Optronics Photoscan 1000), again with a .1 mm resolution on the film. Note that the "footprint" of a LANDSAT picture cell is about 79 x 79 meters [1], compatible with the resolution of typical digital terrain models. The digitized image used for the illustrations in this paper is of lower resolution, however, due to limitations of the optics and electron-optics of the digitizing system. In future studies we intend to use the computer-compatible tapes supplied by EROS [19].

Alignment of real images with terrain models is possible even with low quality image data, but terrain classification using the aligned image and digital surface model requires high quality data.

We generated image output, as for figures 1a, 1b, 7a, and 11, on a drum film-writer (Optronics Photowriter 1500) and interpolated to alleviate undesirable raster effects due to the relatively small number of picture cells in each image.

TRANSFORMATION PARAMETERS

Before we can match the synthetic and the real image, we must determine the nature of the transformation between them. If the real image truly is an orthographic projection obtained by looking straight down, it is possible to describe this transformation as a combination of a translation, a rotation and a scale change. If we use x and y to designate points in the synthetic image and x' and y' for points in the real image, we may write:

$$\begin{bmatrix} x' - x'_0 \\ \\ y' - y'_0 \end{bmatrix} = s \begin{bmatrix} \cos\theta & \sin\theta \\ \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x - x_0 \\ \\ y - y_0 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \\ \Delta y \end{bmatrix}$$

where Δx and Δy are the shifts in x' and y' respectively, θ is the angle of rotation and s is the scale factor. Rotation and scaling take place relative to the centers (x_0,y_0) and (x'_0,y'_0) of the two images in order to better decouple the effects of rotation and scaling from translations. That is, the average shift in x' and y' induced by a change in rotation angle or scale is zero.

In our case, the available terrain model restricts in size the synthetic image. The area over which matching of the two will be performed is thus always fixed by the border of the synthetic image. The geometry of the coordinate transformation is illustrated in figure 8.

CHOICE OF SIMILARITY MEASURE

In order to determine the best set of transformation parameters (Δx, Δy, s, θ), one must be able to measure how closely the images match for a particular choice of parameter values. Let S_{ij} be the intensity of the synthetic image at the i^th picture cell across in the j^th row from the bottom of the image, and define R_{ij} similarly for the real image. Because of the nature of the coordinate transformation, we cannot expect that the point in the real image corresponding to the point (i,j) in the synthetic image will fall precisely on one of the picture cells. Consequently, S_{ij} will have to be compared with R(x',y'), which is interpolated from the array of real image intensities. Here (x',y') is obtained from (i,j) by the transformation described in the previous section.

One measure of difference between the two images may be obtained by summing the absolute values of differences over the whole array. Alternately, one might sum the squares of the differences:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \{S_{ij} - R(x',y')\}^2$$

This measure will be <u>minimal</u> for exact alignment of the images. Expanding the square, one decomposes this result into three terms, the first being the sum of S_{ij}^2, the last the sum of $R^2(x',y')$. The first is constant, since we always use the full synthetic image; the last varies slowly as different regions of the real image are covered. The sum of $S_{ij}R(x',y')$ is interesting since this term varies most rapidly with changes in the transformation. In fact, a very useful measure of the similarity of the two images is the correlation:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} S_{ij}R(x',y').$$

This measure will be <u>maximal</u> when the images are properly aligned. It has the advantage of being relatively insensitive to constant multiplying factors. These may arise in the real image due to changes in the adjustment of the optical or electronic systems.

Note that image intensity is the product of a constant factor which depends on the details of the imaging system (such as the lens opening and the focal length), the intensity of the illumination striking the surface, and the reflectance of the surface. We assume all but the last factor is constant and thus speak interchangeably of changes in surface reflectance and changes in image intensities.

INTERPOLATION SCHEME

The real image intensity at the point (x',y') has to be estimated from the array of known image intensities. If we let $k = \lfloor x' \rfloor$, and $\ell = \lfloor y' \rfloor$ be the integer parts of x' and y', then R(x',y') can be estimated from $R_{k\ell}$, $R_{(k+1)\ell}$, $R_{k(\ell+1)}$ and $R_{(k+1)(\ell+1)}$ by linear interpolation (see figure 9).

$$R_{\ell}(x') = (k+1-x')R_{k\ell} + (x'-k)R_{(k+1)\ell}$$

$$R_{(\ell+1)}(x') = (k+1-x')R_{k(\ell+1)} + (x'-k)R_{(k+1)(\ell+1)}$$

$$R(x',y') = (\ell+1-y')R_{\ell}(x')+(y'-\ell)R_{(\ell+1)}(x')$$

The answer is independent of the order of interpolation and, in fact, corresponds to the result obtained by fitting a polynomial of the form (a + bx' + xy' + dx'y') to the values at the four indicated points. Alignment is not impaired, however, when nearest neighbor interpolation is used instead. This may be a result of the smoothing of the real image as previously described.

CHOICE OF NORMALIZATION METHOD

High output may result as the transformation is changed simply because the region of the real image used happens to have a high average gray-level. Spurious background slopes and false maxima may then result if the raw correlation is used. For this and other reasons, it is convenient to normalize. One approach essentially amounts to dividing each of the two images by its standard deviation; alternately, one can divide the raw correlation by

$$\sqrt{\sum_{i=j}^{n} \sum_{j=1}^{m} S_{ij}^2} \times \sqrt{\sum_{i=j}^{n} \sum_{j=1}^{m} R^2(x',y')}$$

An additional advantage is that a perfect match of the two images now corresponds to a normalized correlation of one. An alternate method uses a normalization factor that is slightly easier to compute and which has certain advantages if the standard deviations of the two images are similar. Instead of using the geometric mean, Hans Moravec proposes the arithmetic mean [20].

$$\left[\sum_{i=1}^{n} \sum_{j=1}^{m} S_{ij}^2 + \sum_{i=1}^{n} \sum_{j=1}^{m} R^2(x',y') \right] /2$$

The first term need not be recomputed, since the full synthetic image is always used. Since we found the alignment procedure insensitive to the choice of normalization method, we used the second in our illustrations.

LOCATING THE BEST MATCH

Now that we have shown how to calculate a good similarity measure, we must find a method to find efficiently the best possible transformation parameters. Exhaustive search is clearly out of the question. Fortunately, the similarity measure allows the use of standard hill-climbing techniques. This is because it tends to vary smoothly with changes in parameters and often is monotonic (at least for small ranges of the parameters).

When images are not seriously misaligned, profiles of the similarity measure usually are unimodal with a well-defined peak when plotted against one of the four parameters of the transformation (see figure 10). It is possible to optimize each parameter in turn, using simple search techniques in one dimension. The process can then be iterated. A few passes of this process typically produce convergence. (More sophisticated schemes could reduce the amount of computation, but were not explored).

When the images are initially not reasonably aligned, more care has to be taken to avoid being trapped by local maxima. Solving this problem using more extensive search leads to prohibitively lengthy computations. We need a way of reducing the cost of comparing images.

USING REDUCED IMAGES

One way to reduce the computation is to use only sub-images or "windows" extracted from the original images. This is useful for fine matching, but is not satisfactory here because of the lack of global context.

Alternately, one might use sampled images obtained by picking one image intensity to represent a small block of image intensities. This is satisfactory as long as the original images are smoothed and do not have any high resolution features. If this is not the case, aliasing due to under-sampling will produce images of poor quality unsuitable for comparisons.

One solution to this dilemma is to low-pass filter the images before sampling. A simple approximation to this process uses averages of small blocks of image intensities. The easiest method involves making one image intensity in the reduced image equal to the average of a 2 x 2 block of intensities in the original image. This technique can be applied repeatedly to produce ever smaller images and has been used in a number of other applications [20,21].

The results of the application of this reduction process to real and synthetic images can be seen in figure 11. First, the most highly reduced image is used to get coarse alignment. In this case extensive search in the parameter space is permissible, since the number of picture cells in the image to be matched is very small. This coarse alignment is then refined using the next larger reduced images (with four times as many picture cells). Finally, the full resolution images are used directly to fine tune the alignment. False local maxima are, fortunately, much rarer with the highly reduced pictures, thus further speeding the search process. It is as if the high resolution features are the ones leading to false local maxima.

We found it best, by the way, to determine good values for the translations first, then rotation and, finally, scale change. Naturally when searching for a peak value as a function of one parameter, the best values found so far for the other parameters are used.

RESULTS OF REGISTRATION EXPERIMENTS

We matched the real and synthetic images using the similarity measure and search technique just described. We tried several combinations of implementation details, and in all cases achieved alignment which corresponded to a very high value of the normalized correlation, very close to that determined manually. For the images shown here, the normalized correlation coefficient reaches .92 for optimum alignment, and the match is such that no features are more than two picture cells from the expected place, with almost all closer than one. (The major errors in position appear to be due to perspective distortion, as described later, with which the process is not designed to cope). The accuracy with which translation, rotation and scaling were determined can be estimated from the above statement.

Overall, the process appears quite successful, even with degraded data and over a wide range of choices of implementation details. Details of interpolation, normalization, search technique, and even the reflectance map do not matter a great deal.

Having stated that alignment can be accurately achieved, we may now ask how similar the real and synthetic images are. There are a number of uninformative numerical ways of answering this question. Graphic illustrations, such as images of the differences between the real and synthetic image, are more easily understood. For example, we plot real image intensity versus synthetic image intensity in figure 12. Although one might expect a straight line of slope one, the scattergram shows clusters of points, some near the expected line, some not.

The cluster of points indicated by the arrow labelled A (figure 12) corresponds chiefly to image points showing cloud or snow cover, with intensity sufficient to saturate the image digitizer. Here the real image intensity exceeds the synthetic image intensity. Arrow B indicates the cluster of points which corresponds to shadowed points. Those near the vertical axis and to its left come from self-shadowed points. Those near the vertical axis and to its left come from self-shadowed surface elements, while those to the right are regions lying inside shadows cast by other portions of the surface. These cast shadows are not simulated in the synthetic image at the moment. Here the synthetic image is brighter than the real image. Finally, the cluster of points indicated by arrow C arises from the valley floor, which covers a fairly large area and has essentially zero gradient. As a result, the synthetic image has constant intensity here, while the real image shows both darker features (such as the river) and brighter ones (such as those due to the cities and vegetation cover). Most of the ground cover in the valley appears to have higher "albedo" than the bare rock which is exposed in the higher regions, as suggested by the position of this cluster above the line of slope one.

If one were to remove these three clusters of points, the remainder would form one elongated cluster with major axis at about 45°. This shows that, while there may not be an accurate point-by-point equality of intensities, there is a high correlation between intensities in the real and synthetic images.

Note, by the way, that no quantization of intensity is apparent in these scattergrams. This is a result of the smoothing applied to the synthetic image and the interpolation used on the real image. Without smoothing, the synthetic image has fairly coarse quantization levels because of the coarse quantization of elevations as indicated earlier. Without interpolation, the real image, too, has fairly coarse quantization due to the image digitization procedure.

Finally, note that we achieve our goal of obtaining accurate alignment. Detailed matching of synthetic and real image intensity is a new problem which can be approached now that the problem of image registration has been solved.

REASONS FOR REMAINING INTENSITY MISMATCHES

We may need more accurate prediction of image intensities for some applications of aligned image intensity and surface gradient information. Thus, it is useful to analyze the reasons for the differences noted between the synthetic and the real image:

Satellite Imaging. Geometric distortion in satellite imagery may be small but noticeable and traceable to several sources [1]. Shifts of several hundred meters can arise. Perspective distortion for the image used here amounted to about 200 meters on the highest peaks, for example.
Intensity distortions are caused by the fact that scan lines are not all sensed by the same sensor [1]. Electronic noise and atmospheric attenuation, dispersion and scattering are also important for some of the spectral bands.

Digitization. When film transparencies are digitized, the resolution limitations of the optics and the nonlinear response of the film are important. More large errors are introduced if an electron-optic device is used. These typically introduce geometric distortions, nonlinearity and nonuniformity of response. Picture cells may not be square and axes not perpendicular.

Terrain Model. Inaccuracies due to manual entry and editing are common in present day digital terrain models. In addition, the contour maps used commonly as source information are already liberally "generalized" and smoothed by the cartographer. Finally, the estimation of surface gradient is likely to be crude, since the data in such maps is intended to be accurate in elevation, not in the partial derivatives of elevation. Coarse quantization of the gradient is one effect of this that has already been mentioned. We hope that terrain models produced by automatic stereo comparators in the future will not suffer from all of these shortcomings.

Reflectance. The assumption of uniform reflectance and the modelling of reflectance by means of the simple, rather ad hoc functions used here contribute errors to the synthetic image. More seriously, cast shadows are not modelled. Illumination from the sky and mutual illumination between mountain slopes are less important. Including even crude surface cover information improves the match between the synthetic and the real image.

Water. In its various forms, water can produce large mismatches since, at least for the shorter wavelengths, moisture in the atmosphere contributes to attenuation and scattering of light. In liquid form, water produces bright, obscuring areas in the form of clouds and dark regions such as rivers and lakes. Snow and ice provide highly reflective areas which produce large mismatches.

In view of all these factors, it is surprising that a match as good as that in figure 12 is possible.

FURTHER IMPROVEMENT OF THE SYNTHETIC IMAGE

Using the original digital tapes [19] would eliminate the errors we believe are due to the digitization process. Most of the geometric distortion can be dealt with as well [1]. Further match improvement must come from better synthetic images.

The most significant step here would be the inclusion of surface cover information. Even a coarse categorization into materials of grossly differing "albedo" might be useful. Conversely, of course, one can exploit the difference in intensities between the real and the synthetic image to estimate surface reflectance. Since alignment is possible without accurate reflectance models, the ratio of real to synthetic intensity (a measure akin to albedo) can be used in terrain classification, particularly if it is calculated for each of the spectral bands.

Cast shadows are fairly easy to deal with, if we implement a hidden-surface algorithm to determine which surface elements can be seen from the source. This computation can be done fairly quickly using a well known algorithm [22]. Sky illumination in shadowed areas presents no great stumbling block in this regard.

The quality of terrain models is likely to increase most rapidly when fully automatic scanning stereo comparators become available. Until then, hand-editing of hand-traced information will have to be used to limit the errors in the estimation of gradient.

One notion that shows great promise is that of masks derived from both the terrain model and the real image. The masks are used to limit the correlation operation to those areas which are not as likely to lead to mismatches. Areas of very high intensity in the image, for example, may suggest cloud or snow cover, and ought not to be used in the matching operation. Similarly, it may be that areas of certain elevations and surface gradients are better than others for matching. The correlation can be improved considerably if we use only those regions above the elevations at which dense vegetation exists and below the elevation at which snow may have accumulated. A slightly more sophisticated method would note that snow tends to remain longer on north-facing slopes.

THE INFLUENCE OF SUN ELEVATION

Aerial or satellite photographs obtained when the sun is low in the sky show the surface topography most clearly. In this case, the surface gradient is the major factor in determining surface reflectance. Ridges and valleys stand out in stark relief, and one gets an immediate impression of the shape of portions of the surface. Conversely, variations in surface cover tend to be most important when the sun is high in the sky. Photographs obtained under such conditions are difficult to align with a topographic map -- at least for a beginner.

What is the sun elevation for which these two effects are about equally important? Finding this value will allow us to separate the imaging situations into two classes: those which are more suited for determining topography and those which are more conducive to terrain classification success. We will use a simple model of surface reflectance. Suppose that the surface has materials varying in "albedo" between ρ_1 and ρ_2. Next, suppose that the surface slopes are all less than or equal to tan(e). The incident angles will vary between e - (90° - ϕ) and e + (90° - ϕ), where ϕ is the elevation of the sun. If we use the same simple reflectance function employed before, we find that for the two influences on reflectance to be just equal:

$$\rho_1 \cos(e + 90° - \phi) = \rho_2 \cos(e - 90° + \phi)$$

Expanding the cosine and rearranging this equation leads to:

$$\tan(\phi) = \left|\frac{\rho_1 + \rho_2}{\rho_1 - \rho_2}\right| \tan(e)$$

When, for example, the surface materials have reflectances covering a range of two to one and the sun elevation is 35°, then regions with surface slopes above approximately 0.23 ($e \approx 13°$) will have image intensities affected more by surface gradient than by surface cover. Conversely, flatter surfaces will result in images more affected by variations in surface cover than by the area's topography.

One possible conclusion is that alignment of images with terrain models is feasible without detailed knowledge of the surface materials if the sun elevation is small and the surface slopes are high. Since LANDSAT images are taken at about 9:30 local solar time [1], the first condition is satisfied and alignment of these images is possible even in only lightly undulating terrain.

Conversely, if one is attempting terrain classification in anything but flat regions, high sun elevations are needed. Curiously, LANDSAT does not provide such imagery despite the fact that one of its main applications is in land use classification.

SUMMARY AND CONCLUSIONS

We have seen that real images can be aligned with surface models using synthetic images as an intermediary. This process works well despite many factors which contribute to differences between the real and synthetic images. The computations, while lengthy, are straightforward, and reduced images have been used to speed up the search for the best set of transformation parameters.

Several applications of aligned images and surface information have been presented. More can be found; for problems in a different domain, see reference [23] for example. Aside from change detection, passive navigation, photo-interpretation, and inspection of industrial parts, perhaps the most important application lies in the area of terrain classification.

So far, no account has been taken of the effect of varying surface gradient, sun position, and reflective properties of ground cover. Recently, some interest has arisen in an understanding of how surface layers reflect light [24, 25, 26] and how this understanding might aid the interpretation of satellite imagery [27, 28, 29].

It is imperative that interpretation of image formation be guided by an understanding of the imaging process. This, in turn, can be achieved if one understands how light is reflected from various surfaces and how this might be affected by such factors as light source position, moisture content and point in the growth cycle of vegetation.

ACKNOWLEDGEMENTS

We would like to thank Professor Kurt Brassel (State University of New York, Buffalo, New York), Professor Thomas Peucker (Simon Fraser University, Burnaby, B.C., Canada), Dr. Bala (EIKONICS, Burlington, Massachusetts), George E. Lukes (U.S.A. E.T.L., Fort Belvoir, Virginia), and Dr. Harry Barrow (Stanford Research Institute, Palo Alto, California) for generously supplying digital terrain models.

We would also like to express our appreciation to Blenda Horn, Eva Kampits, Karen Prendergast and Patrick Winston who provided numerous helpful suggestions and improved the presentation of the results. Special thanks are due Al Paone and Russ Clark of the Massachusetts Institute of Technology Graphic Arts Service for help in the presentation of the photographic reproductions.

REFERENCES

1. R. Bernstein, "Digital image processing of earth observation sensor data," IBM Journal of Research and Development, Vol. 20, pp. 40-57.

2. B. K. P. Horn, "Obtaining shape from shading information," in P. H. Winston (ed.), The Psychology of Computer Vision, New York, McGraw-Hill, 1975, pp. 115-155.

3. B. K. P. Horn, "Understanding image intensity," Artificial Intelligence, Vol. 8, pp. 201-230, April 1977.

4. K. Brassel, Modelle und Versuch zur automatischen Schraglicht-schattierung, 7250 Klosters, Switzerland, Buchdruckerei E. Brassel, 1973.

5. B. K. P. Horn, "Automatic hill-shading using the reflectance map," unpublished, 1976.

6. Carte nationale de la Suisse, Swiss National Tourist Office, New York, New York 10020.

7. T. K. Peucker, "Computer cartography," Resource paper no. 17, Association of American Geographers, Washington, D.C. 1972.

8. Bala, "Experimental heterodyne optical correlator," ETL-0071, U.S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia 22060, October 1976.

9. "AS-11B-X automated stereo mapper," RADC-TR-76-100, Rome Air Development, Griffiths Air Force Base, New York 13441, April 1976.

10. "N.C.I.C. digital terrain file," and "N.C.I.C. 1:250 000-scale digital terrain library," National Cartographic Information Center, U.S. Geological Survey, Reston, Virginia 22092

11. "The nautical almanac for the year 1972," U.S. Government Printing Office, Washington, D.C. 20402.

12. "The American ephemeris and nautical almanac for the year 1972," U.S. Government Printing Office, Washington, D.C. 20402.

13. "Explanatory supplement to the astronomical ephemeris and the American ephemeris and nautical almanac," Her Majesty's Stationery Office, London, England 1961.

14. Simon Newcomb, A Compendium of Spherical Astronomy, New York, McMillan & Co., 1985, 1906.

15. W. M. Smart, Textbook on spherical astronomy, Cambridge, Cambridge University Press, 1931, 1965.

16. E. W. Woolard & G. M. Clemance, Spherical Astronomy, Cambridge, Cambridge University Press, 1966.

17. B. K. P. Horn, "Celestial navigation suite for a programmable calculator," unpublished, 1976.

18. B. Hapke, "An improved theoretical lunar photometric function," The Astronomical Journal, Vol. 71, pp. 333-339.

19. "Computer compatible tape (CCT) index," EROS Data Center, Sioux Falls, South Dakota 57198.

20. H. P. Moravec, "Techniques towards automatic visual obstacle avoidance," Proceedings of the Fifth International Conference on Artificial Intelligence, Cambridge, MA, 1977, pp. 584.

21. S. L. Tanimoto, "Pictorial feature distortion in a pyramid," Computer Graphics and Information Processing, Vol. 5, pp. 333-353, September 1976.

22. I. E. Sutherland, R. F. Sproul & R. A. Schumacker, "A characterization of ten hidden-surface algorithms, Computing Surveys, Vol. 6, pp. 1-55.

23. D. Nitzan, A. E. Brain & R. O. Duda, "The measurement and use of registered reflectance and range data in scene analysis," Proc. of the IEEE, Vol. 65, pp. 206-220.

24. E. L. Simmons, "Particle model theory of diffuse reflectance: effect of non-uniform particle size," Applied Optics, Vol. 15, pp. 603-604, 1976.

25. W. W. M. Wendlandt & H. G. Hecht, Reflectance Spectroscopy, New York, Interscience Publishers, 1976.

26. P. Oetking, "Photometric studies of diffusely reflecting surfaces with applications to the brightness of the moon," Journal of Geophysical Research, Vol. 71, pp. 2505-2514, May 1966.

27. C. J. Tucker & M. W. Garret, "Leaf optical system modeled as a stochastic process, Applied Optics, Vol. 16, pp. 635-642, March 1977.

28. C. J. Tucker, "Asymptotic nature of grass canopy spectral reflectance," Applied Optics, Vol. 16, pp. 1151-1156, May 1977.

29. M. J. Duggin, "Likely effects of solar elevation on the quantification of changes in vegetation with maturity using sequential LANDSAT images. Applied Optics, Vol. 16, pp. 521-523, March 1977.

LAMBERT DENT_DE_MORCLES

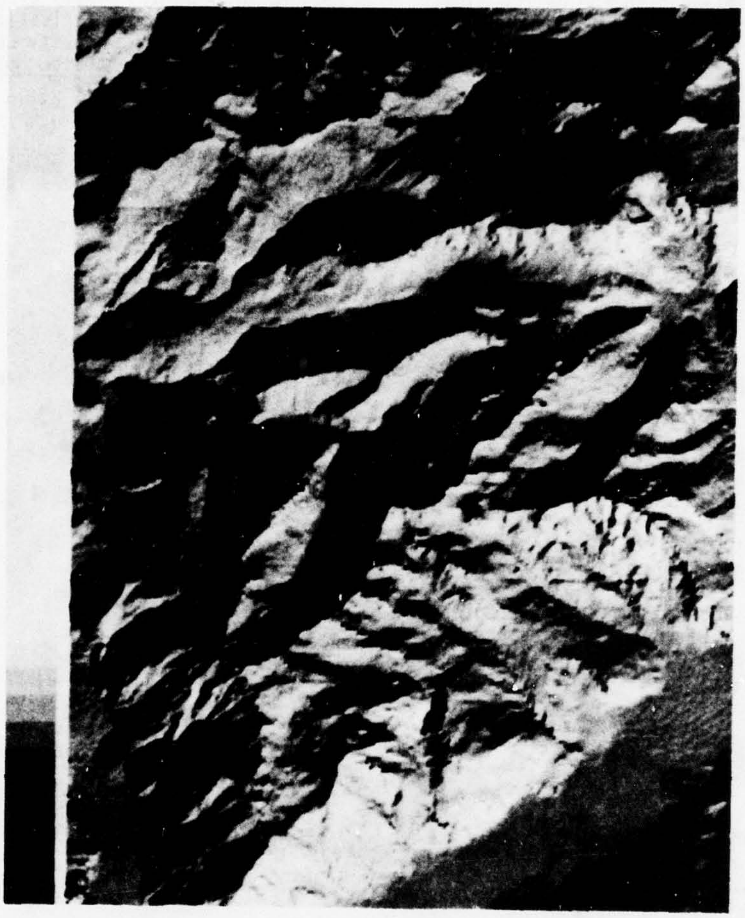


IMAGE2.2 at photowriter resolution 3.
Data linearly interpolated. JULY 14, 1977.

Figure 1a. Early morning (9:52 G.M.T.) synthetic image.

" LAMBERT DENT_DE_MORCLES

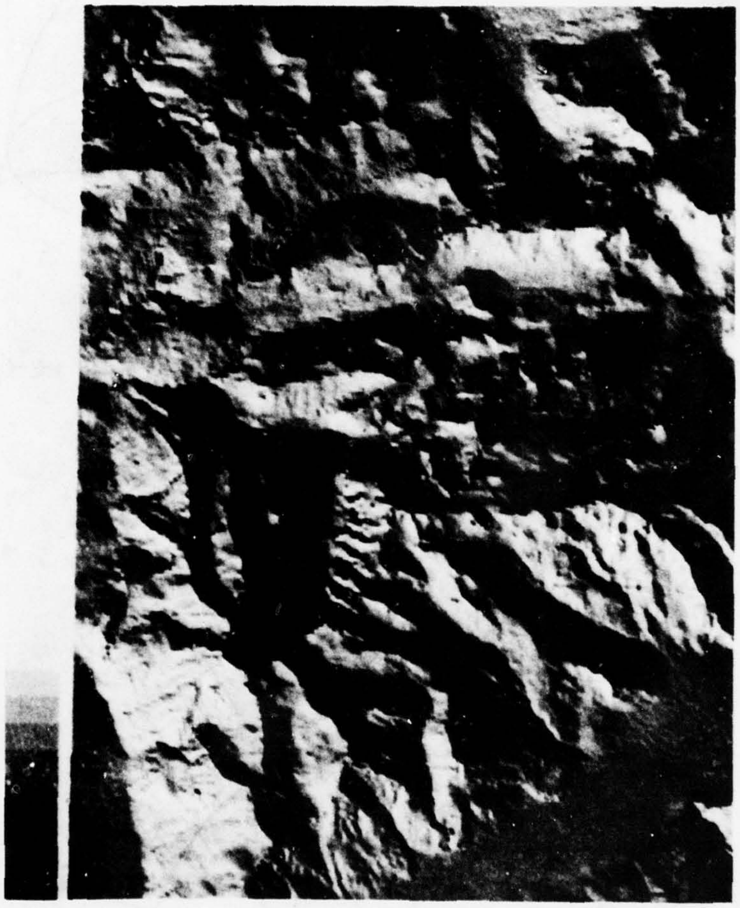


IMAGE2.4. second exposure. Resolution 3.
data interpolated. Date 15 JULY 1977.

Figure 1b. Early afternoon (13:48 G.M.T.) synthetic image.

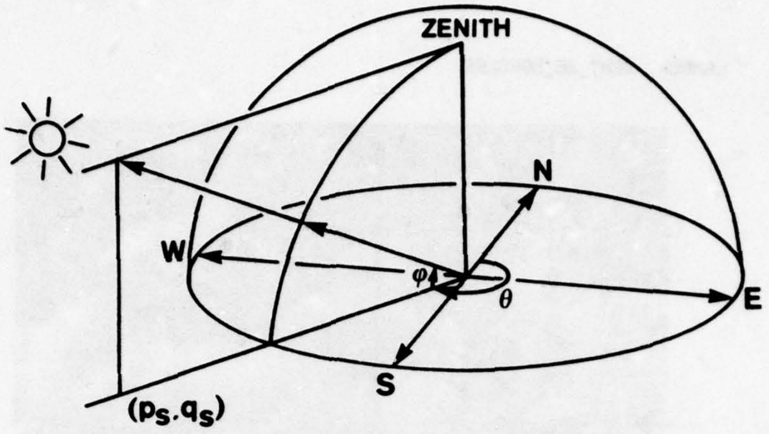


Figure 2. Definition of azimuth and elevation of the sun.

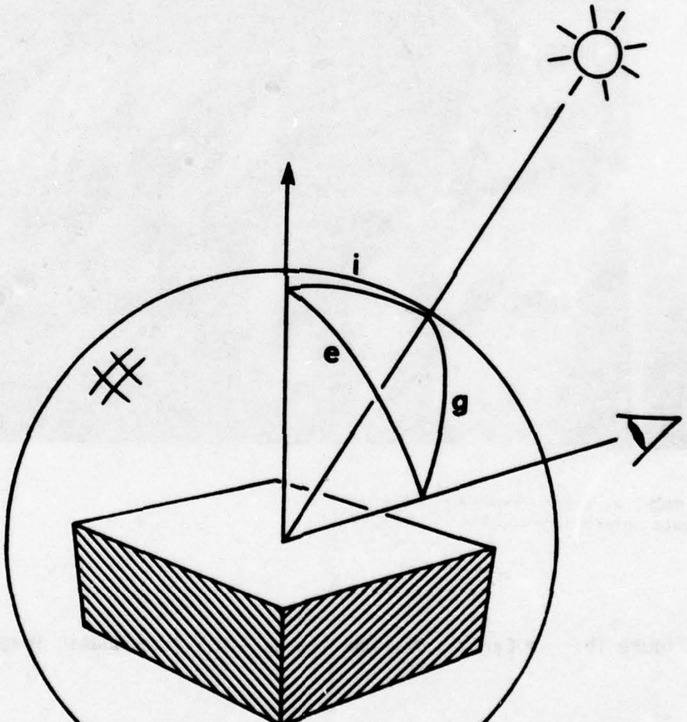


Figure 3. The geometry of light reflection from a surface element is governed by the incident angle, i, the emittance angle, e, and the phase angle, g.

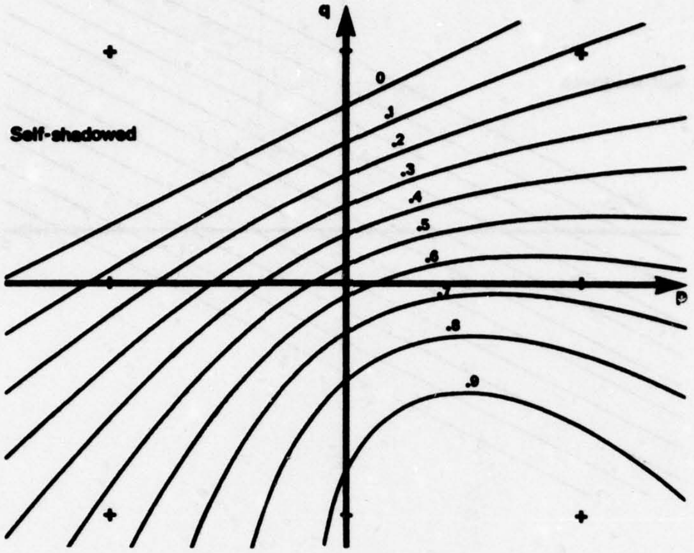


Figure 4a. Reflectance map used in the synthesis of figure 1a. The curves shown are contours of constant $\phi_1(p,q)$ for $\rho = 1$.

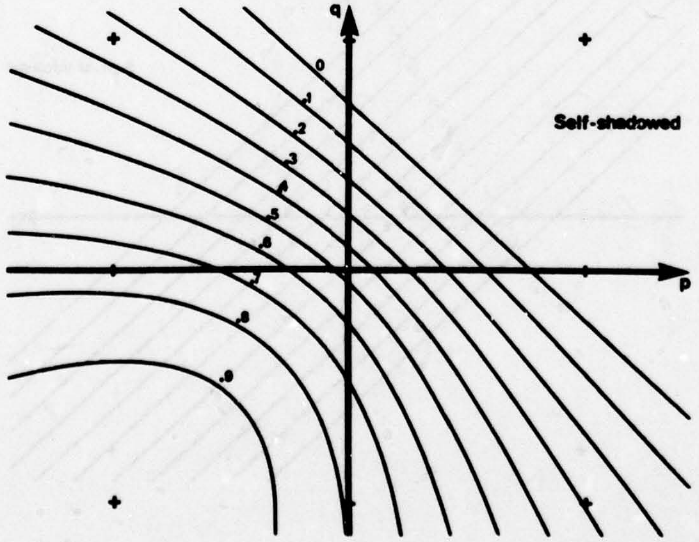


Figure 4b. Reflectance map used in the synthesis of figure 1b.

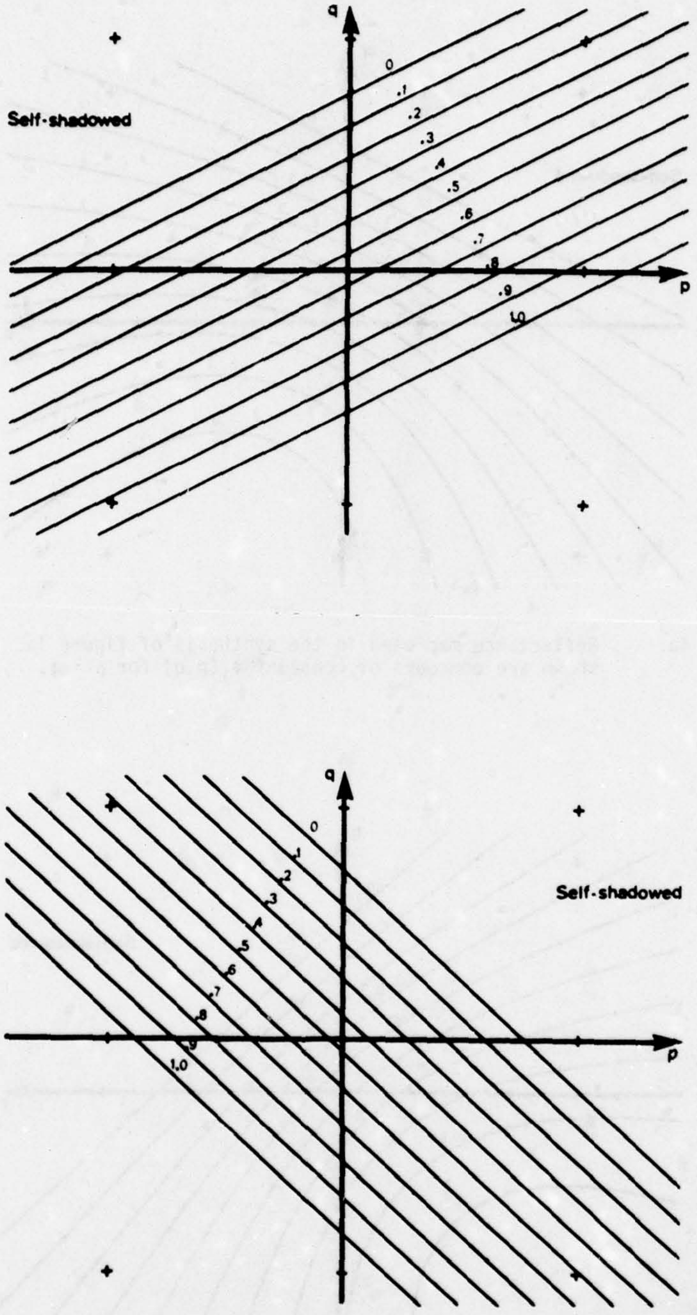


Figure 5. Alternate reflectance map, which could have been used in place of the one shown in figure 4a. The curves shown are contours of constant $\phi_2(p,q)$ for $\rho = 1$.

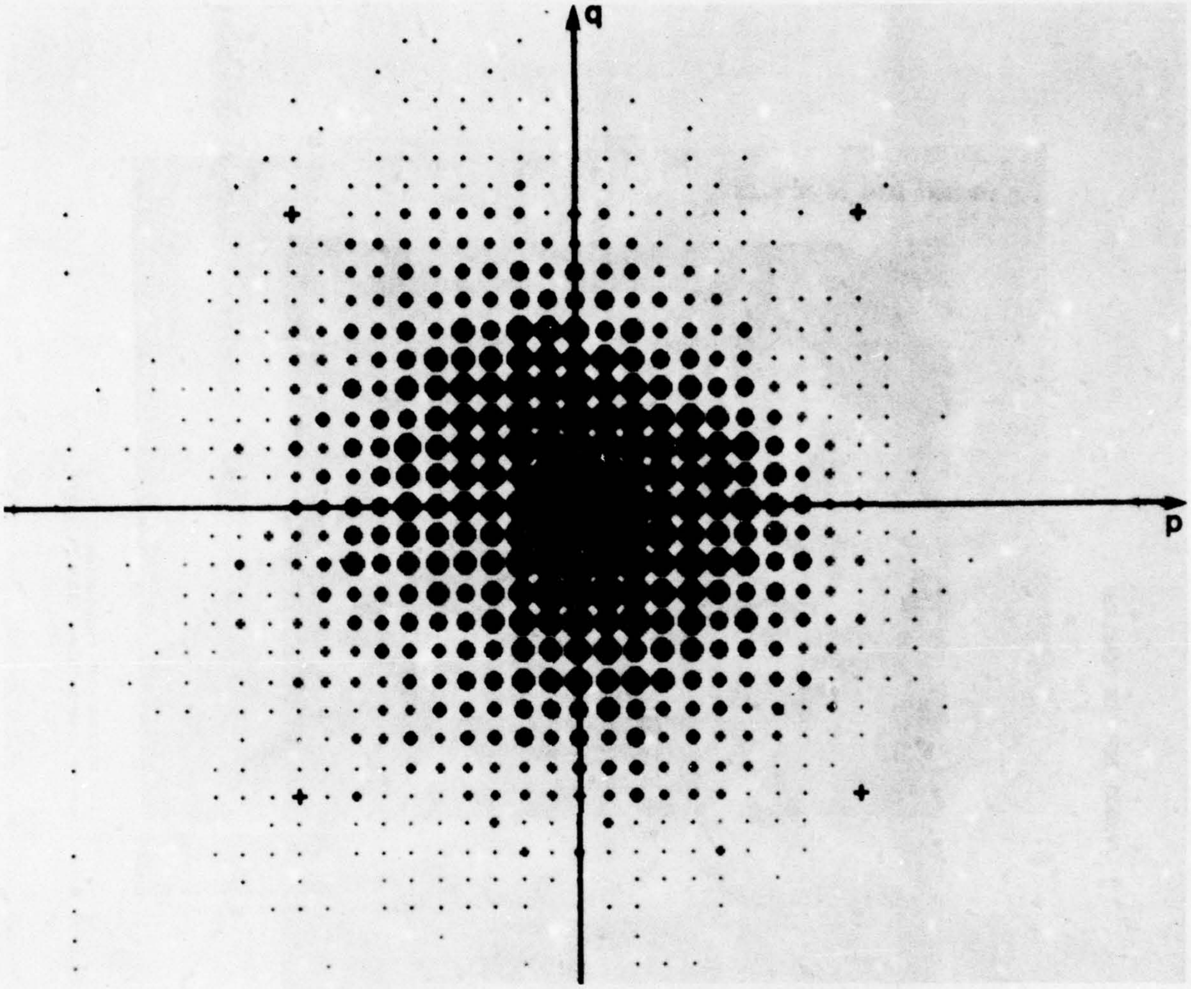


Figure 6. Scattergram of surface gradients found in the digital terrain model.

LAMBERT DENT_DE_MORCLES

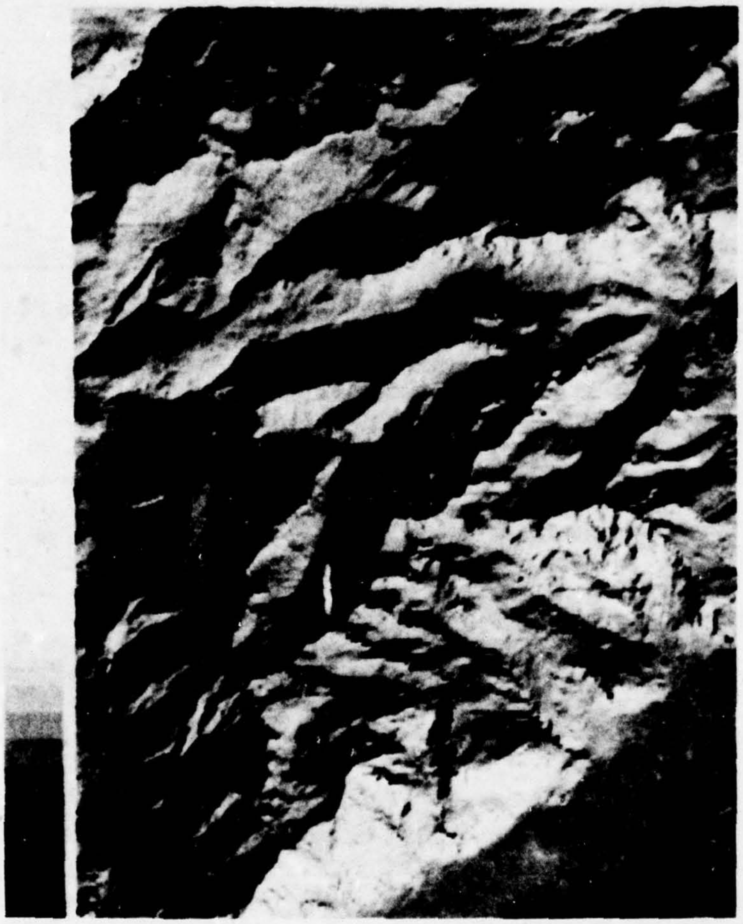


IMAGE2.2 at photowriter resolution 3.
Data linearly interpolated. JULY 14, 1977.

Figure 7a. Synthetic image used in the alignment experiments.

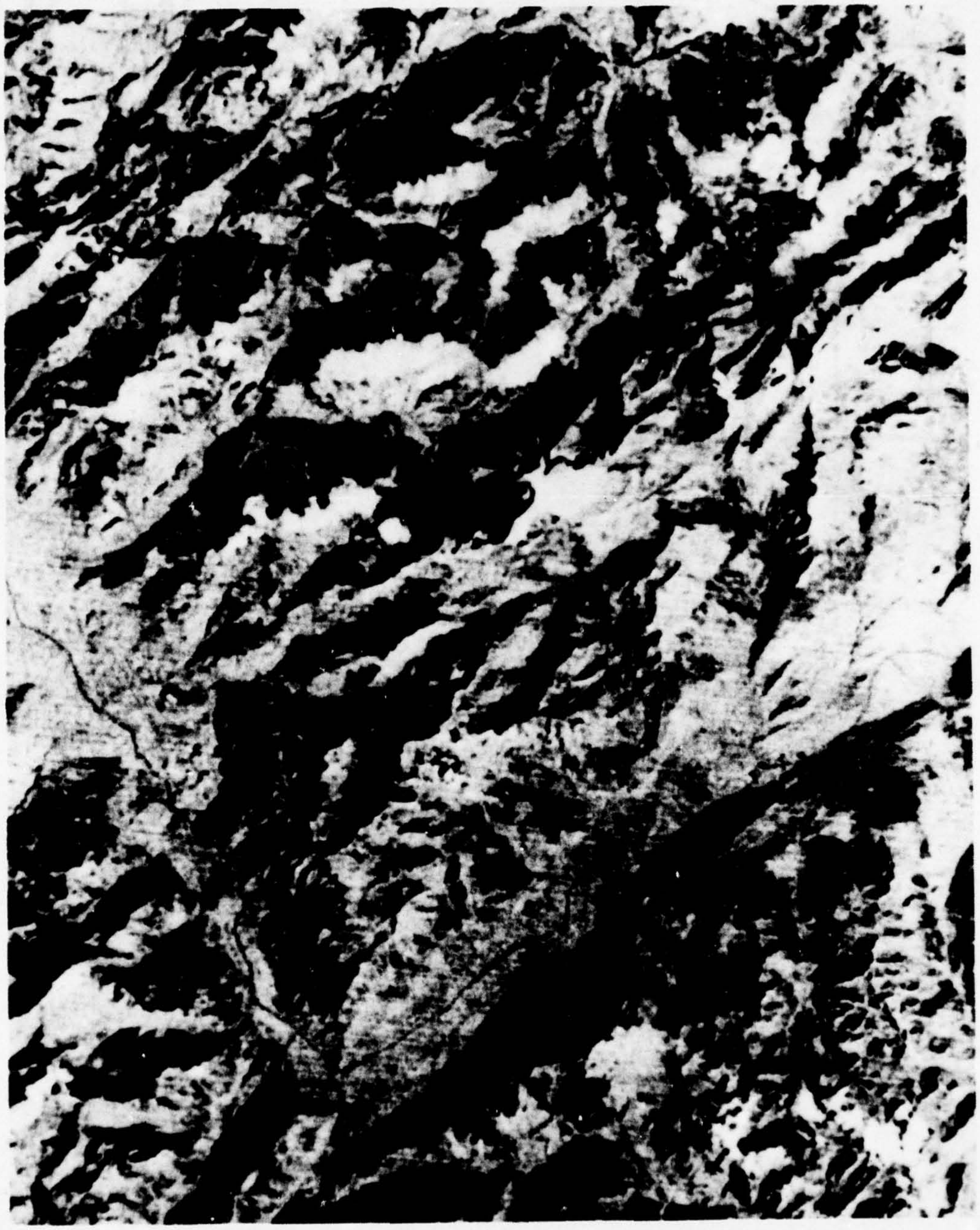


Figure 7b. Enlargement of the transparency containing the real image used in the alignment experiments.

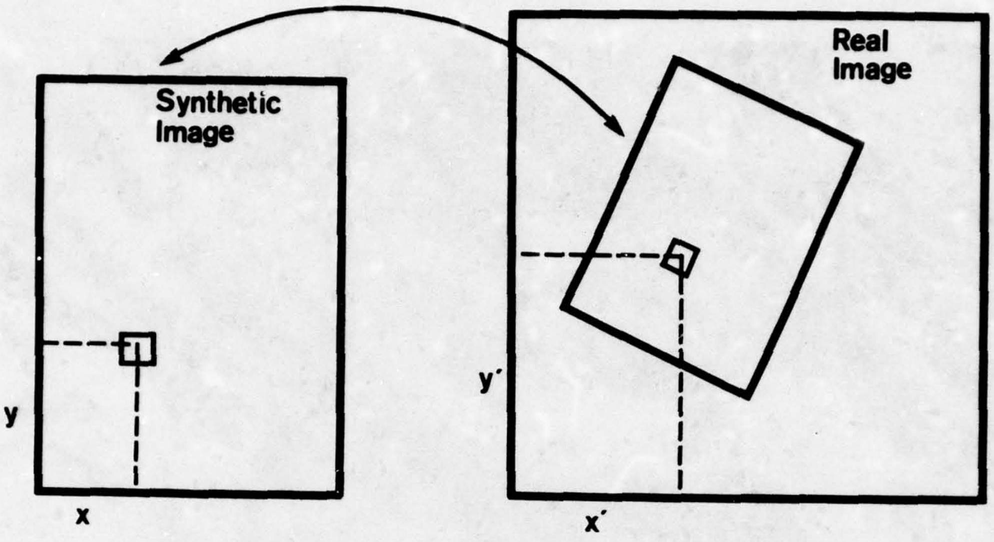


Figure 8. Coordinate transformation from synthetic image to real image.

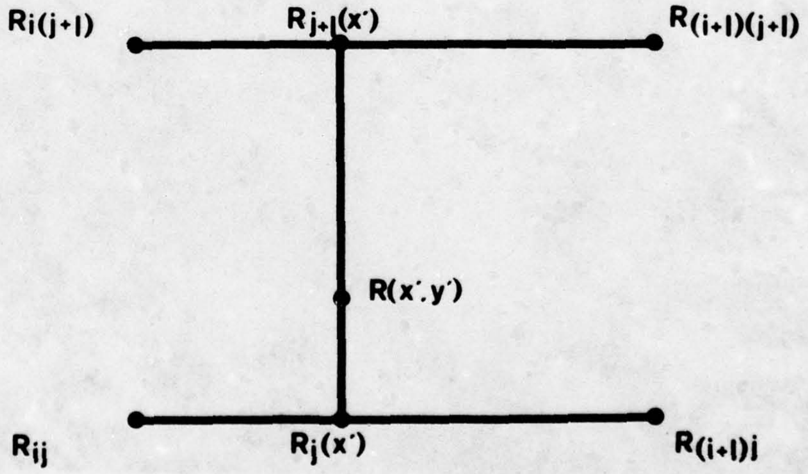
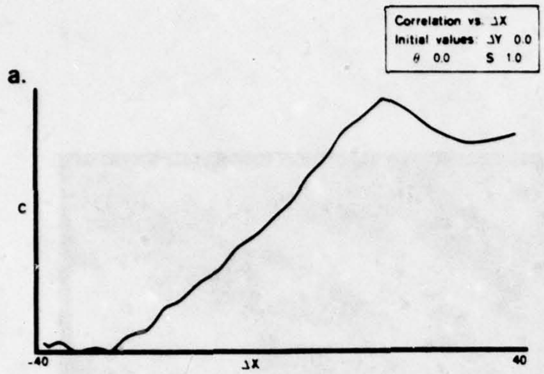


Figure 9. Simple interpolation scheme applied to the real image array.



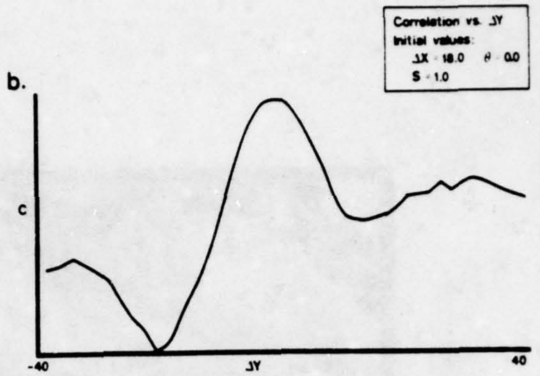
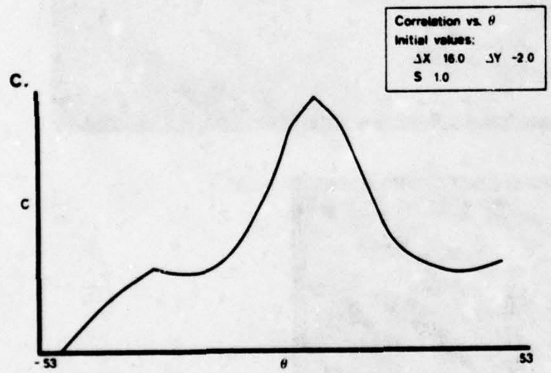


Figure 10a. Variation of similarity measure with translation in x direction.

Figure 10b. Variation of similarity measure with translation in y direction.



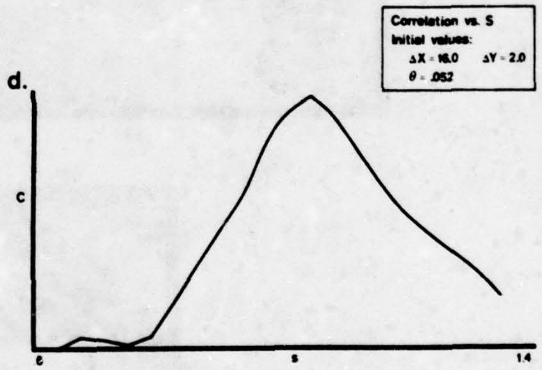


Figure 10c. Variation of similarity measure with rotation.

Figure 10d. Variation of similarity measure with scale changes.

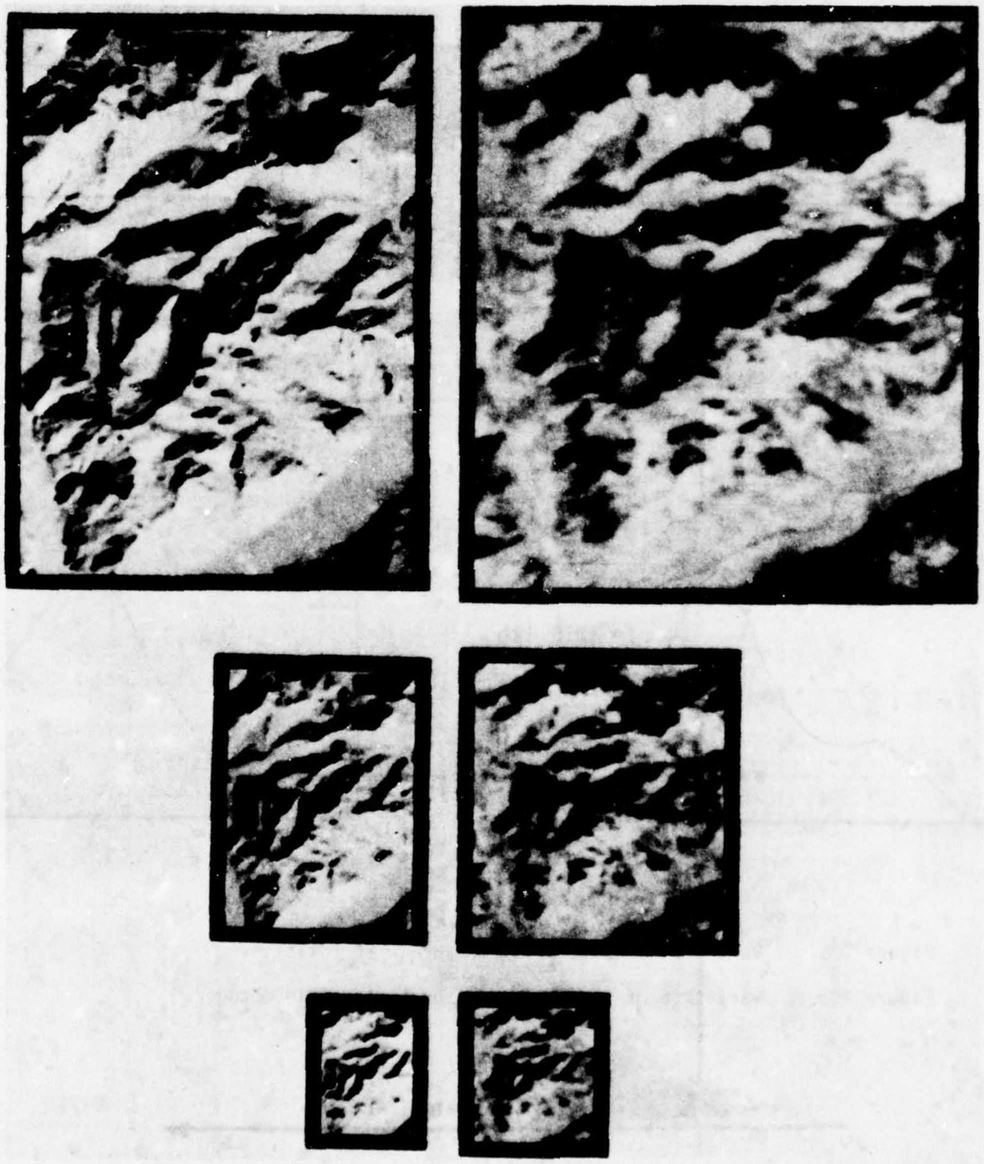


Figure 11. Successive reduction by factors of two applied to both the synthetic (left) and the real (right) image.

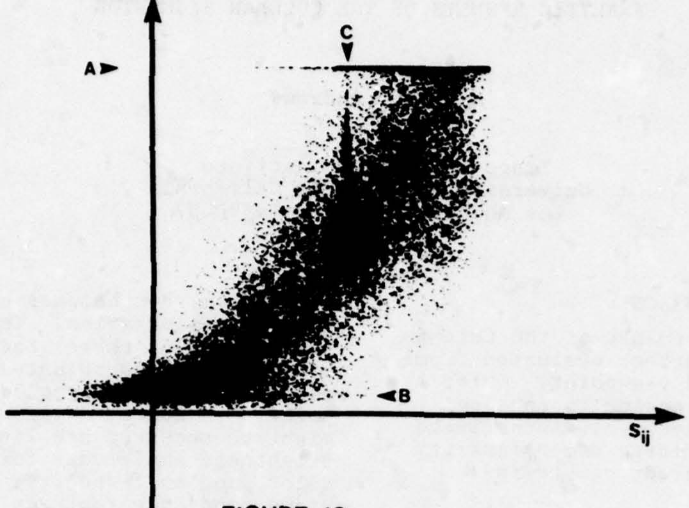


FIGURE 12a

FIGURE 12b

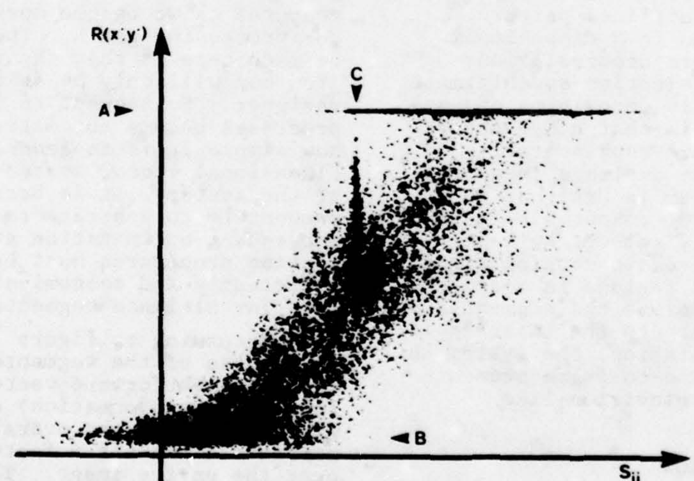


Figure 12a. Scattergram of real image intensities versus synthetic image intensities based on $\phi_1(p,q)$.

Figure 12b. Scattergram of real image intensities versus synthetic image intensities based on $\phi_2(p,q)$.

ANALYTIC RESULTS OF THE COLEMAN SEGMENTOR

Harry C. Andrews

Image Processing Institute
University of Southern California
Los Angeles, California 90007

ABSTRACT

Clustering performance of the Coleman segmentor has been further evaluated from a pattern recognition viewpoint. Quite effective feature rejection to enhance tighter, more homogeneous clusters (image segments) results in large dimensionality reduction with equivalent clustering performance.

INTRODUCTION

Automatic bottom up human unassisted image segmentation has been developed by Coleman [1] for the Image Understanding program. The system utilizes pattern recognition techniques in N dimensional vector space to perform decorrelation, clustering, feature rejection and ultimate segmentation. The only underlying assumption for the process is that homogeneous clusters in N space are representative of homogeneous regions of an image in perceptual space. The system is designed to operate with any set of computable features and will automatically select the best subset of those features to develop tightly clustered homogeneous regions in N space which then serve to define the segmentation of the original image. In the interest of smart sensor implementation, the system has been designed for frame-to-frame segmentation for real time television-like sensors.

SEGMENTOR CONFIGURATION

Figure 1 presents a block diagram representative of the system design of the segmentor. The first component of the system is the "feature computation" phase This process computes the features that the designer feels will be relevant for effective clustering. Essentially, features are computed up to as high a resolution as at every pixel if desired. Because the features to be computed are defined by the user, it is at this phase that human intuitive and design processes are brought to bear on the segmentation problems. Once the human defined features are computed, the system then becomes automatic for subsequent optimization. The computed features (N of these) then define an N dimensional coordinate system wherein each pixel will subsequently represent a point in N space. Typical features that might be computed are listed in table 1 Brightness amplitudes for monochrome, color, and multispectral scenes are obvious candidate features. Texture features might be delineated by edges or other spatial frequency processors and are listed in the table. Finally, nonlinear spatial filtering processes might also be useful for segmentation and this class of features is listed as well. Obviously, as humans we can continue to generate more features as we become more familiar with our processing goals. The only point to be made here is that the feature computation box will only be as clever as its designer. Subsequent to this phase, all processes become automatic. However, note how simple it is to generate fairly large dimensional vector spaces at the front end of the system. It is because of man's propensity to generate so much data that subsequent optimization and feature rejection procedures must be developed to efficiently and economically process such data for ultimate segmentation purposes.

Returning to figure 1 we see that the next phase of the segmentor configuration is a straightforward vector space rotation (unitary transformation) defined by the eigenvectors of the overall covariance matrix between all N features computed over the entire image. The objective of this phase is to decorrelate the features such that clustering is implemented in N dimensional decorrelated space. In this way good features can be selected individually and bad features rejected individually without concern as to correlation properties with other features. This will allow efficient compaction of good clustering features into a few parameters thereby providing a large dimensionality reduction. However it is important to realize that feature reduction does not occur immediately following the rotation process but only subsequent to clustering analysis.

This brings us to the next step in the system which is a k-means clustering algorithm in N dimensional rotated space. This algorithm converges to a set of k-mean points describing the best assignment of pixel features to k-clusters such that the sum of within cluster distances is the smallest. The disadvantage of the algorithm is that it requires knowledge of the number of clusters, k, in advance. Clearly this is unknown and consequently the k-means clustering routine must be implemented for all reasonable values of k (i.e. k=1,...,16). Subsequent blocks in the figure are designed to determine the best number of cluster and the best features to provide the tightest cluster distributions.

Once the k-means cluster algorithm has converged to the minimum spread of points in N space, a fidelity measure, β, is computed to establish the tightness of the points within the clusters and the degree of spread or separateness of the clusters one from another. This fidelity measure is given by

$$\beta(k) = \mathrm{tr}[S_w(k)]\ \mathrm{tr}[S_b(k)]$$

where $[S_w(k)]$ is the within cluster scatter matrix and $[S_b(k)]$ is the between cluster scatter matrix [2]. It can be shown that β is everywhere nonnegative, has at least one maximum, and achieves that maximum where the ratio of the within cluster scatter equals the between cluster scatter. Therefore it is hypothesized that the optimal number of clusters (k) occurs at β equal to its maximum. Therefore these values of β and k are used to control the output segmentor and the feature rejector.

The feature rejector provides the function of removing those features which do not contribute to tight homogeneous clusters. Consequently, this process borrows from supervised pattern recognition theory in which feature selection/rejection is often implemented through the use of the Bhattacharyya distance function [3]. This function provides a measure of the usefulness of a particular dimension or feature by investigating that feature's ability to separate the data points into the proper clusters determined by the k-means convergence algorithm. This measure is provided by mean and variance parameters determined by each dimension for all the clusters. Those features or dimensions which do not provide well-defined clusters (due to separate means and tight variances) are rejected, thereby leaving good features for more tightly homogeneous clusters.

EXPERIMENTAL RESULTS

A variety of images have been segmented using the above clustering algorithm with varying degrees of perceptual success. Figures 2 and 3 present these results in pictorial form. Figure 2a and 3d were original monochrome images while figure 2d was a color image and figure 3a was a ten band multispectral image. Various clustering results are presented for each image for viewer inspection. The last sequence in figure 3 represents clustering on frame-to-frame imagery to illustrate the potential for real time hardware smart sensor implementation.

Probably a more relevant representation of the segmentor in operation is to view the Bhattacharyya measures and clustering fidelity factors all as a function of k, the number of clusters for each iteration of the k-means clustering algorithm. These results are presented in figures 4 and 5. In figure 4 two plots are presented illustrating the performance of the Bhattacharyya feature rejector. In figure 4a the Bhattacharyya distance values are plotted for each dimension or feature in the correlated space for the variables $\{x_1, x_2, \ldots, x_N\}$ from figure 1. In figure 4b the Bhattacharyya distances are plotted for each rotated dimension or feature in the decorrelated space for the variable $\{y_1, y_2, \ldots, y_N\}$ of figure 1. It is immediately obvious that by decorrelating (rotating the space) one outstanding feature results which hopefully will allow effective clustering in a vastly reduced vector space (see figure 2b). In addition it is obvious that the good features (large Bhattacharyya values), tend to be good for all cluster numbers indicating a degree of consistency which allows feature rejection of those dimensions with small Bhattacharyya measure with some degree of confidence.

Figure 5 indicates how the cluster fidelity parameter, β, behaves as the number of clusters increases. Specifically, figure 5a indicates that for the monochrome APC image, without feature rejection, the peak of β is quite poorly defined because of the presence of a lot of useless features essentially adding noise to the well-defined clusters. However for the case of the four best features or the single best feature, a much more marked peak results at a lower cluster number. A similar effect occurs for the colored house of figure 5b. However from the curves of all features compared to the best few features, not as dramatic a change occurs. This is because the use of color features provides a considerable improvement in the segmentation power of the system compared to having only monochrome features. This result correlates well with our intuitive experiences in which color and multispectral signatures provide quite useful aids for human visual segmentation procedures.

CONCLUSION

The above description covers the highlights of the segmentor developed by Coleman. The interested reader is referred to reference [1] for details of the system. The algorithm represents a bottom up attempt at automatically segmenting imagery without the aid of human intervention. It allows any conceivable set of features to be used for clustering but reserves the right to feature reject those parameters which do not contribute to well-defined, tight clusters. The technique is based upon the principles of mathematical clustering algorithms in N dimensional vector space. The underlying hypothesis for success of the technique is based upon the premise that tight, well-defined, homogeneous clusters in vector space correspond to well-defined homogeneous regions in an image. If this premise is true, successful (i.e. consistent with human perception) segmentation results. If unsuccessful segmentation results, then improper features are provided in the feature computation phase which, through the linear operations of decorrelation and feature rejections, do not provide proper region segments. It is then conjectured that nonlinear transformations (or other features) are necessary. Finally, the segmentor has been designed with smart sensor real time implementation in mind. The hardware construction of such a system is under contemplation.

REFERENCES

1. G. Coleman, "Image Segmentation by Clustering," University of Southern California, USCIPI Report 750, August 1977.

2. For the rather involved details of this fidelity measure and its derivation, the interested reader is referred to reference [1] above.

3. H.C. Andrews, Introduction to Mathematical Techniques in Pattern Recognition, Wiley-Interscience, New York, New York, 1972.

FEATURE TABLE 1

INDEX	FEATURE DESCRIPTION	FEATURE CLASS
x_1	monochrome brightness	monochromatic amplitude
x_2	red color brightness	color amplitude
x_3	green color brightness	
x_4	blue color brightness	
x_5	band 1 brightness	multispectral amplitude
x_6		
⋮		
x_{10}	band 6 brightness	
x_{11}	Sobel magnitude on x_1	texture feature
x_{12}	Sobel magnitude on x_2	
⋮		
x_{20}	Sobel magnitude on x_{10}	
x_{21}	Sobel phase on x_1	texture orientation
⋮		
x_{30}	Sobel phase on x_{10}	
x_{31}	mode filter on x_1	nonlinearly filtered feature
⋮		
x_{40}	mode filter on x_{10}	
x_{41}	dispersion filter on x_1	
⋮		
x_{50}	dispersion filter on x_{10}	

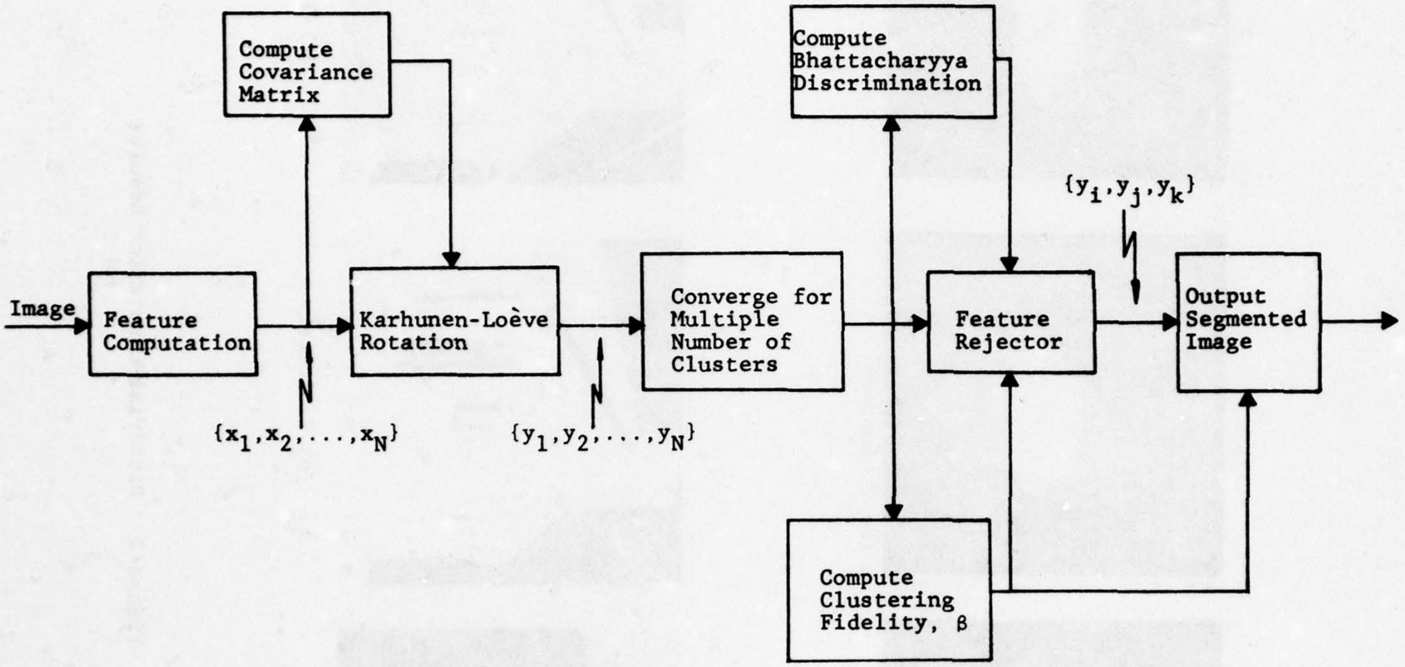
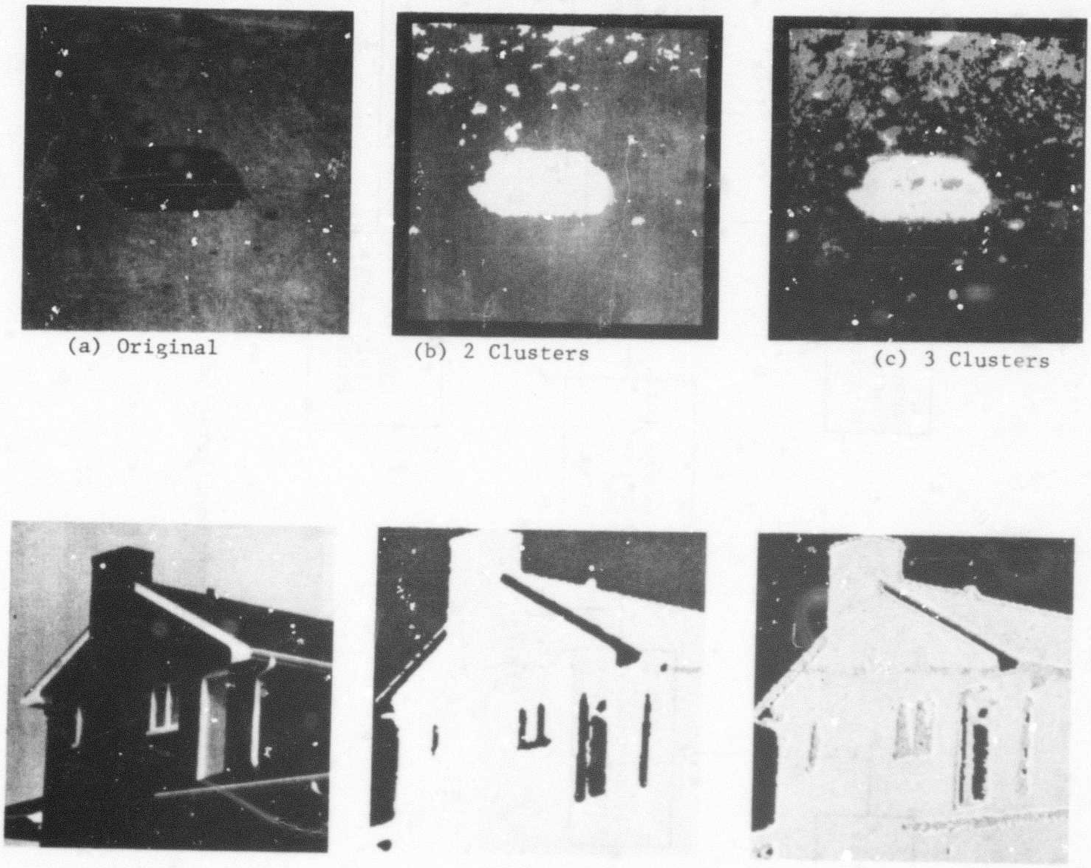


Image
Feature Computation
Compute Covariance Matrix
Karhunen-Loève Rotation
$\{x_1, x_2, \ldots, x_N\}$
Converge for Multiple Number of Clusters
$\{y_1, y_2, \ldots, y_N\}$
Compute Bhattacharyya Discrimination
Feature Rejector
Compute Clustering Fidelity, β
$\{y_i, y_j, y_k\}$
Output Segmented Image

Figure 1. Segmentor Configuration



(a) Original

(b) 2 Clusters

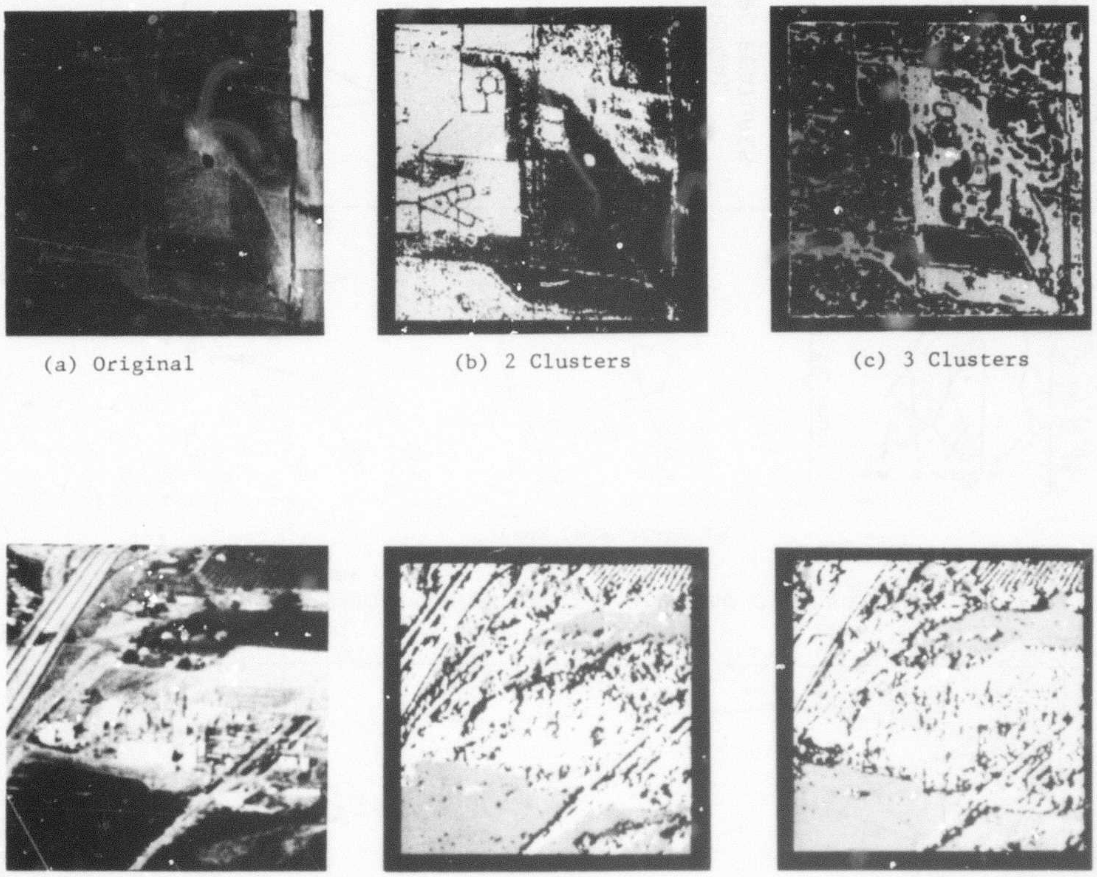
(c) 3 Clusters

(d) Original

(e) 2 Clusters

(f) 3 Clusters

Figure 2. Pictorial Clustering Results



(a) Original

(b) 2 Clusters

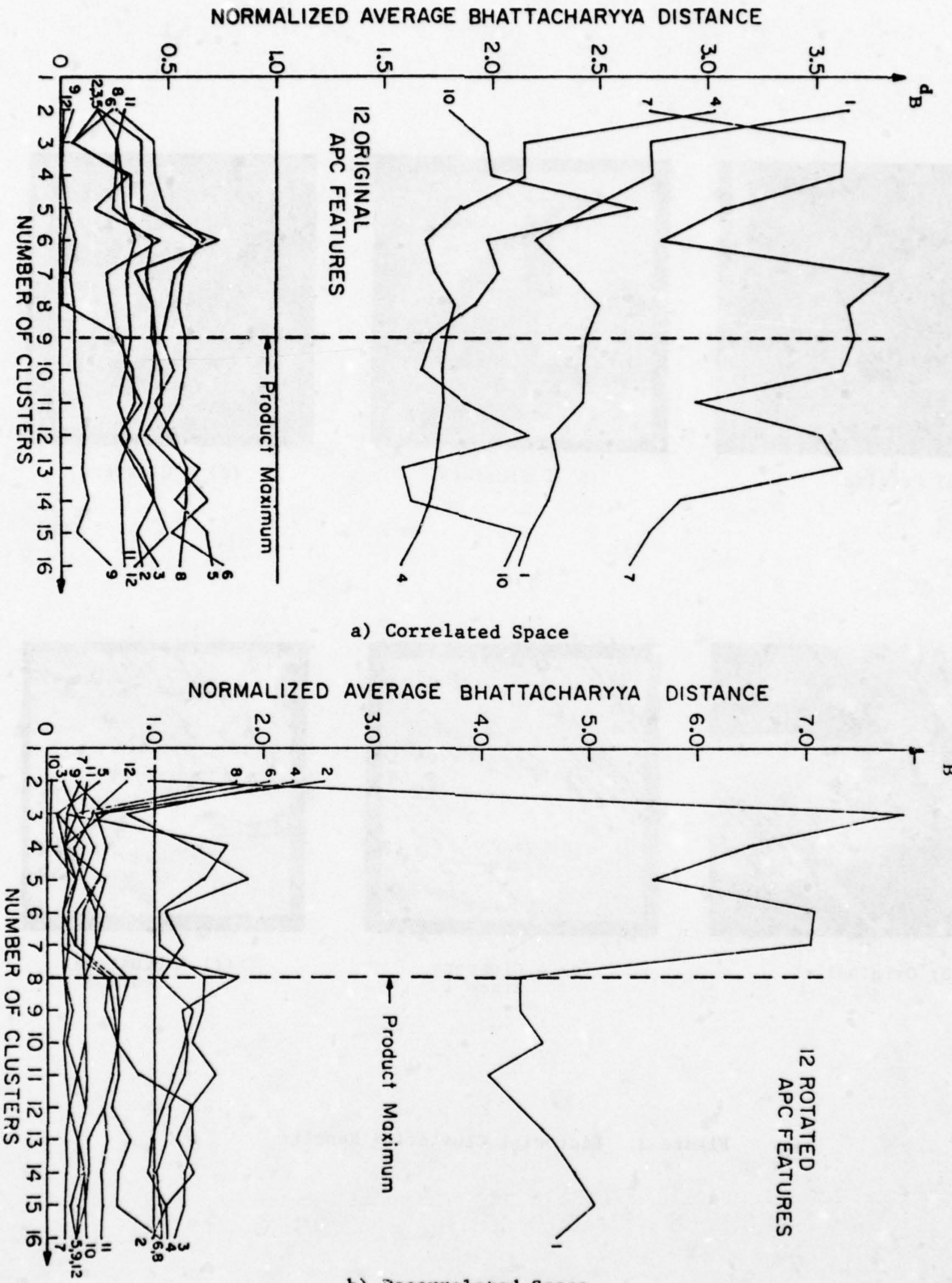
(c) 3 Clusters

(d) Original

(e) 4 Clusters
frame 1

(f) 4 Clusters
frame 5

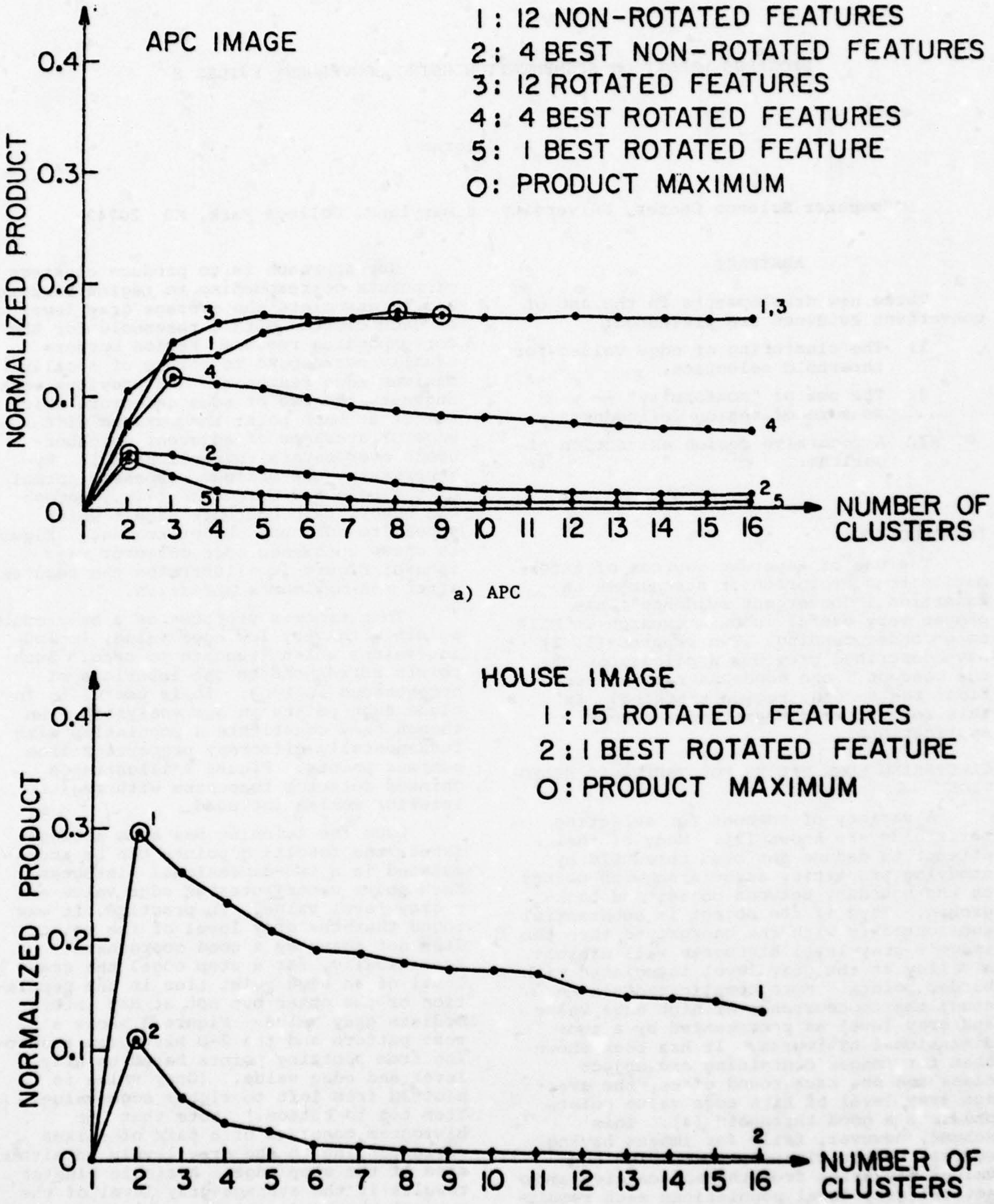
Figure 3. Pictorial Clustering Results



a) Correlated Space

b) Decorrelated Space

Figure 4. Bhattacharyya Feature Rejector for the APC Image.



a) APC

b) Colored House

Figure 5. Clustering Fidelity Measure.

PROGRESS REPORT ON SEGMENTATION USING CONVERGENT EVIDENCE

D. L. Milgram

Computer Science Center, University of Maryland, College Park, MD 20742

ABSTRACT

Three new developments in the use of convergent evidence are presented:

1. The clustering of edge values for threshold selection.
2. The use of "conformity" -- a measure of region definedness.
3. A recursive region extraction algorithm.

INTRODUCTION

The use of separate sources of information to corroborate or strengthen an assertion ("convergent evidence") has proven very useful in our research on FLIR image understanding. Two reports [1, 2] have described previous applications of the concept: one concerns region extraction; the second, region tracking. In this report, we discuss three recent applications.

CLUSTERING EDGE VALUES FOR THRESHOLD SELECTION

A variety of schemes for selecting thresholds are known [3]. Many of them attempt to deduce the best threshold by studying properties associated with points on the boundary between object and background. Thus if the object is substantial and contrasts with the background, then the image's gray level histogram will exhibit a valley at the gray level associated with border points. More complicated schemes study the cooccurrence of high edge value and gray level as represented by a two-dimensional histogram. It has been shown that for images containing one object class and one background class, the average gray level of high edge value points predicts a good threshold [4]. This scheme, however, fails for images having several object classes, since high edge values may arise from the adjacencies among several gray level populations each requiring a different threshold. In what follows, we present an approach to thresholding in a multi-population environment.

Our approach is to produce clusters of points corresponding to region borders and to associate the average gray level of each cluster with a threshold for the corresponding region. Region borders usually correspond to points of locally maximum edge response. Our previous work suggests the use of edge detectors which select at each point the maximum difference of averages of adjacent neighborhoods over several directions [5]. By suppressing non-maximum responses normal to the selected direction (i.e., across the edge), thin contours result which appear to surround object regions. Figure 1b shows unthinned edge detector response; Figure 1c illustrates the results after non-maximum suppression.

This process produces as a by-product points with very low edge value, including values which truncate to zero. Such points correspond to the interiors of homogeneous regions. It is useful to include such points in our analysis, even though they constitute a population with fundamentally different properties from contour points. Figure 2 illustrates thinned detector responses with region interior maxima included.

Once the thinning has been accomplished, the resulting points can be accumulated in a two-dimensional histogram. Each point contributes an edge value and a gray level value. In practice, it was found that the gray level of the point does not serve as a good coordinate. Specifically, for a step edge, the gray level of an edge point lies in one population or the other but not at any intermediate gray value. Figure 3 shows a test pattern and the 2-D histogram resulting from plotting points based on gray level and edge value. (Gray value is plotted from left to right; edge value, from top to bottom.) Note that the histogram consists of a pair of spikes corresponding to the gray levels on either side of the step edge. A single cluster results if the average gray level of the two neighborhoods which contributed the maximal edge response at the point is plotted on the gray scale. Figure 3c shows the effect on the histogram of using the average gray level instead of the point gray level.

In images consisting of disjoint homogeneous objects on a homogeneous background, the thinned edge contours will correspond to clusters in the 2-D histogram. Moreover, their interiors will produce clusters at edge values close to zero. It is important to note that the size of a cluster (i.e., the number of points in it) is closely related to properties of the region it describes. Thus interior clusters relate both to the area of the region and to the size of the neighborhood over which the local operations (edge detection, non-maximum suppression) are defined. For small object regions, there may be no points sufficiently far from the object boundary to resist suppression. Thus interior clusters may be indistinguishable from noise, or may be non-existent.

Clusters of points at higher edge values are more likely to be significant (based on our homogeneity assumptions). The size of an edge cluster is therefore related to the perimeter of the surrounded region in the image. Since perimeter increases (roughly, for digital images) as the square root of area, the edge clusters for objects of moderately different areas should, nonetheless, be of comparable size. A priori estimates of size are of use in discriminating true edge clusters from random noise.

Edge clusters and interior clusters bear certain relationships to one another. For example, an edge cluster whose centroid is (e,g), where e is the average edge value and g the average plotted gray level, probably serves as the contour separating two regions of gray level g-e/2 and g+e/2. Finding two interior clusters at gray levels g-e/2, g+e/2, respectively, would serve as confirmation to this assertion. Conversely, to determine whether two regions with average gray level g_1, g_2 respectively, share a common edge (i.e., are adjacent), one could attempt to locate an edge cluster with centroid $(|g_2-g_1|, (g_1+g_2)/2)$. Figure 4 shows three regions with various types of adjacency and the 2-D histograms which derive from them. Lower gray level values correspond to darker shades of gray.

We have investigated some simple methods of cluster extraction, and we now describe one which has been moderately successful. First, use the histogram of thinned edge values (the projection of the 2-D histogram on the edge axis) to detect edge value ranges containing significant peaks. Each of these ranges corresponds to a strip across the 2-D histogram. Now construct a gray value histogram for each strip and segment it according to its peaks. Each such segment corresponds to a rectangle in the 2-D histogram. Clusters are associated with well-populated rectangles. Thresholds are then computed as the average gray levels within clusters.

Given a set of thresholds for an image, it is unclear how one applies them to extract regions. For example, consider the drawing in Figure 5a and its 2-D histogram, Figure 5b. The center of the edge cluster belonging to the interior clusters at gray levels 30 and 40 is at gray level 35; while the edge cluster separating the interior clusters at 20 and 40 has 30 as its center. Thus the thresholds are 30 and 35. The threshold at 30 will optimally separate the background from the outer boundary of the ring; however, it will cause the hole in the ring to break up in a random fashion. The threshold at 35 will in fact separate the hole from the ring but will assign too many border points of the 20-40 border to the background region. Thus neither threshold is by itself optimal. One possible solution is to partition the 2-D histogram into disjoint regions which are labelled as to object class (Figure 6). Thus all points in the original image will be classified based on a feature pair (gray level and edge value). The location of each feature pair in the partitioned histogram will determine the object class to which each image point belongs. Methods for partitioning the 2-D histogram are being investigated.

THE USE OF "CONFORMITY" - A MEASURE OF REGION DEFINEDNESS

The "Superslice" algorithm [1] relies on the heuristic that thresholded object regions are distinct from background because they contrast with their surround at a well-defined border. The coincidence of high contrast and high edge value at the border of a thresholded region is an example of the use of convergent evidence supporting the assertion of the object region. The definedness of the border may be evaluated as the percentage of the border points which coincided with the location of thinned edge (locally maximum edge response). Thus a match score of 50% means that half the border points are accounted for as being on the edge. However, it does not mean that the matched points adequately represent the object. Figure 7 illustrates two cases of 50% match. (Matched points are indicated by thick strokes.) Clearly, the second case is a better representation than the first.

The traveral of the border of a thresholded region induces an ordering on the matched points. Let $r_1, \ldots, r_n$ be the runs of matched points encountered during a border traversal. By connecting the proximal ends of runs along the traversal, one creates a polygonal approximation to the thresholded region. We define "conformity" as the measure of match of the polygonal approximation to the thresholded region. High conformity means that the region is well-represented by its approximation regardless of the actual percentage of matched border points. Figure 7a illustrates low conformity; while Figure 7b shows good conformity.

Conformity is evaluated as the ratio of the absolute difference in area (between the two polygonal representations) to the area of the threshold region. Experiments have indicated its utility as a feature discriminating noise from objects. A quantitative study of its classification value is underway.

A RECURSIVE REGION EXTRACTION ALGORITHM

Much effort has been devoted to algorithms which segment images based on regions homogeneous with respect to selected feature values. Ohlander [6] developed a recursive framework for region extraction which asserts the existence of homogeneous regions based on well-defined modes in one of several histograms. The extraction of a set of points corresponding to a mode of some feature forces the recomputation of the other feature histograms for the remaining points. Given a sufficient number of (more or less) independent features, point sets can be continually extracted until further decomposition produces only noise regions.

In our work, we have attempted to extract only those regions whose assertion is warranted by additional evidence. We have developed measures of contrast and well-definedness in order to accept or reject regions proposed by slicing. These heuristics can be used to strengthen the recursive algorithm and to allow its operation when very few feature histograms are present. In fact, using gray level as the single feature for computing a histogram, it is possible to segment complex images.

The algorithm operates as follows:

a. Choose a "best" slice range from the histogram. (Section 1 suggests some possibilities.)

b. Slice the image according to the slice range.

c. Extract those regions which satisfy the Superslice criteria (see Section 2 for further details). Return all other points in the slice range to the background set, and recompute the feature histogram.

d. Apply the algorithm recursively to the background set and to each of the extracted regions.

The use of the Superslice algorithm to extract object regions while rejecting noise regions means that points lying in the slice range but not conforming to an object region are returned to the pool of histogrammed points. Thus a "liberal" slice range (i.e., extending beyond valleys in the histogram) will allow good region definition with the extra points returned to the unclassified pool. The resulting histogram does not have a "carved-out" look to it, and further slice range selection is possible.

In order to test out this idea, we constructed an interactive system which allows the user to display the gray level histogram, select a slice range, extract well-defined regions, and construct masks of accepted regions. Figure 8 displays the products of the analysis of an image using this approach.

REFERENCES

1. D. L. Milgram, Region extraction using convergent evidence, Proceedings: ARPA Image Understanding Workshop, April, 1977, 58-64.

2. D. L. Milgram, Region tracking using dynamic programming, TR-539, Computer Science Center, University of Maryland, College Park, MD, May 1977.

3. A. Rosenfeld and A. C. Kak, Digital Picture Processing, Academic Press, New York, 1976.

4. J. S. Weszka, R. N. Nagel and A. Rosenfeld, A threshold selection technique, IEEE-TC-23, 1974, 1322-1326.

5. Algorithms and Hardware Technology for Image Recognition, First Semi-Annual Report on Contract DAAG53-76-C-0138, Computer Science Center, University of Maryland, College Park, MD, October 1976.

6. R. Ohlander, Analysis of Natural Scenes, Ph.D. Thesis, Carnegie-Mellon University, Pittsburgh, PA, December 1976.

Figure 1a. Window containing tank.
1b. Edge detector response (thresholded).
1c. Thinned edge detector response.

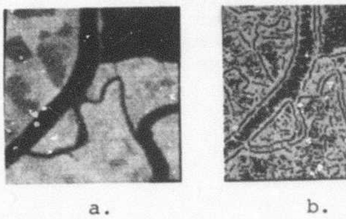


Figure 2a. LANDSAT window of Monterey.
2b. Thinned edge detector response.

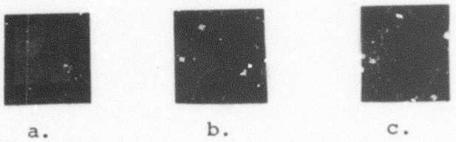


Figure 3a. Square on background.
3b. 2-D histogram with gray level as x-axis and edge value as y-axis (stretched).
3c. 2-D histogram with average gray level as x-axis and edge value as y-axis (stretched).

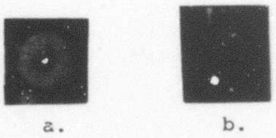


Figure 4a. Disk (gray level 30) within ring (gray level 40) within background (gray level 20).
4b. 2-D histogram (distorted for visibility - interior of background is leftmost, topmost cluster).

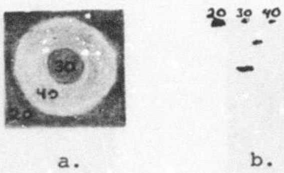


Figure 5a. Adjacent object region on background (same as Figure 4a).
5b. 2-D histogram.

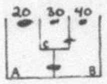


Figure 6. 2-D histogram of Figure 5a, partitioned into classification regions.

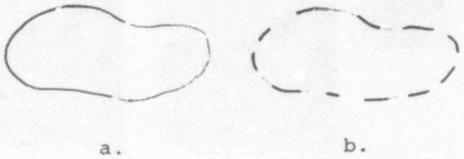


Figure 7a. Contour whose matched edge points (thickened strokes) exhibit poor conformity.
7b. Contour showing good conformity.

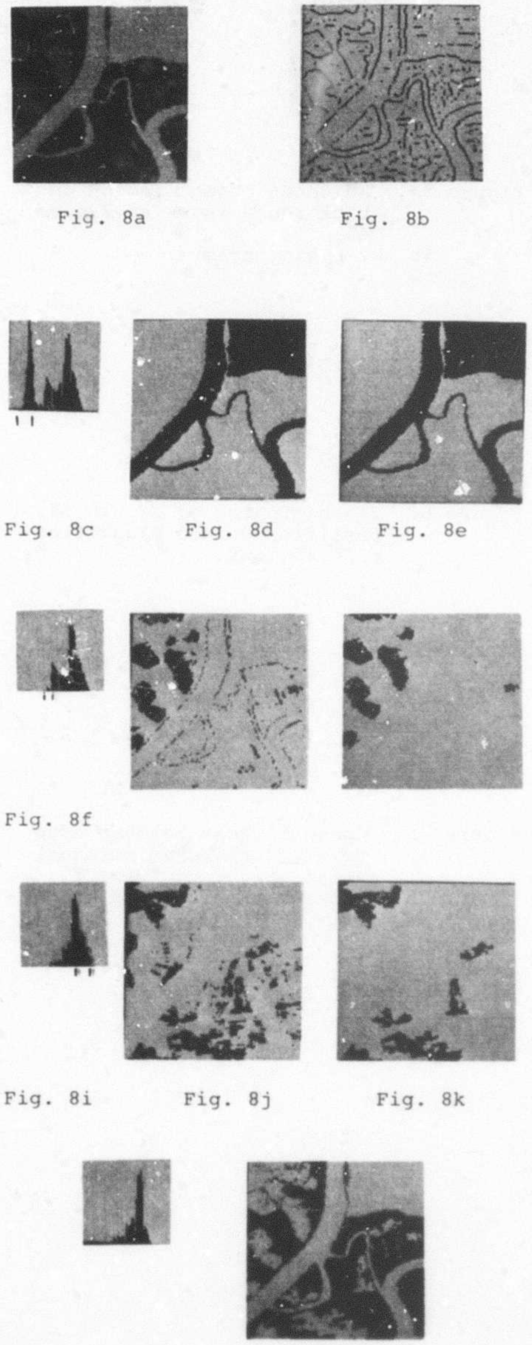


Fig. 8a Fig. 8b

Fig. 8c Fig. 8d Fig. 8e

Fig. 8f

Fig. 8i Fig. 8j Fig. 8k

Fia. 8L Fig. 8m

Figure 8. Interactive region analysis.

a. Monterey ERTS window.
b. Edge map.
c. Histogram of Figure 8a, with selected slice range indicated.
d. Map of slice range. Within range points are black.
e. Map of Superslice regions extracted from slice range
f. Histogram of points remaining after deleting extracted regions.
g. Map of next slice range.
h. Map of extracted regions.
i. Histogram of remaining points.
j. Map of next slice range.
k. Map of extracted regions.
l. Histogram of remaining points.
m. Gray level image of remaining points.

SESSION V

PROGRAM REVIEWS

BY

PRINCIPAL INVESTIGATORS

IMAGE UNDERSTANDING AT USC: APRIL - SEPTEMBER 1977

Harry C. Andrews

Image Processing Institute
University of Southern California
Los Angeles, California 90007

ABSTRACT

The past six months have seen progress on a variety of fronts including object estimation in noise, phase unwrapping for restoration, perceptual models for smart sensor design, and degree of freedom models for a variety of SAR imaging configurations. These and other results will be summarized.

SMART SENSORS

Testing of the Sobel circuit and adaptive circuit is progressing quite nicely at the Hughes Research Laboratories. Results on these two circuits are discussed elsewhere in this workshop proceedings for the interested reader. Plans for subsequent circuit designs include: a) circuits operating at real time TV rates with 8 bits (½%) accuracy, b) development of a texture measuring circuit, and c) investigation of a segmentor circuit.

IMAGE UNDERSTANDING PROJECTS

Professor Nevatia and Dr. Price have concentrated on applying image understanding techniques to locating structures in aerial images. Contextual knowledge is utilized in the process. In addition, circle detectors in noisy images have been investigated for application to shape structure development algorithms.

Professor Pratt and his students have concentrated on quantitative edge evaluators with some quite successful comparisons resulting. Signal detection as well as pattern recognition approaches are utilized in the evaluation process. In addition, other results include initiation of research into the use of the singular value decomposition operator as a useful texture aid. So far, various texture patterns have been analyzed, and the singular value map of both artificial and natural textures of similar classes (i.e. grass) are nearly identical while for different classes (i.e. ivy), the maps are also different. Therefore, it appears that the shape of the singular values may provide a useful indication of texture measure.

Finally, the Coleman segmentor has been refined and used to process monochrome, color, multispectral, and frame-to-frame imagery with varying degrees of "perceptual success." This workshop proceedings has a summary of the segmentor performance evaluation parameters, and the interested reader may learn of the details of the project from USCIPI Report #750.

PERCEPTUAL MODELING FOR IMAGE UNDERSTANDING

Two topics in the area of human visual system (HVS) modeling appear to be quite relevant to the image understanding program. First as a preprocessor, it now appears feasible to use the nonlinear perceptual model as a front end transformation preprocessor on a smart sensor device. This will allow subsequent signal and image understanding processing to be implemented much more efficiently in an adaptively compressed image domain. Secondly the statistical and potential transform domain properties of this perceptual space are analyzed and results presented, indicating quite high potential for efficient color image coding and texture evaluation and measurement.

RESTORATION

Digital image restoration has been a research topic for some time now and has resulted in a variety of reconstruction algorithms. Recently progress has been made in the area of detection-estimation theory applied to boundary definitions of objects buried in large sensor noise systems. This work has culminated in a dissertation topic and the interested reader is referred to USCIPI Report #760 for further details.

The problem of a posteriori (after the fact) restoration has been under investigation with particular emphasis on recovering the phase of the optical transfer function (OTF) distortion of a space invariant imaging system. Recursive algorithms for both magnitude (modulation transfer function or MTF) and phase of the

OTF have been developed. Real imagery is now being processed with significant results.

Finally a new project has been initiated over the past six months specifically designed to meet the image understanding needs of nonvisual imagery. Specifically, the degrees of freedom of a variety of radar imaging systems are under investigation. The stripping and spotlight SAR modes of radar imagery collection are being analyzed as to their inherent information content, both to define and measure limitations and provide suggested improvements. However the more relevant objective of the project is to provide automatic interpretation and understanding of the imagery. This is particularly appropriate as human interpretation of radar imagery is somewhat limited by the foreign looking appearance of such pictures compared to natural (visibly sensed) imagery.

INTERACTIVE AIDS FOR CARTOGRAPHY AND PHOTO INTERPRETATION: PROGRESS REPORT, OCTOBER 1977.

H.G. Barrow (Principal Investigator),
R.C. Bolles, T.D. Garvey, J.H. Kremers,
K. Lantz*, J.M. Tenenbaum, and H.C. Wolf.

Artificial Intelligence Center,
SRI International
Menlo Park, CA 94025.

I INTRODUCTION

This report describes the ongoing SRI image understanding project. The central scientific goal of this project is to investigate and develop ways in which diverse sources of knowledge may be brought to bear on the problem of interpreting images. The research is focused on the specific problems entailed in interpreting aerial photographs for cartographic or intelligence purposes. Additional details are to be found in earlier progress reports [1] [2] [3].

A key concept is the use of a generalized digital map to guide the process of image interpretation. This map is actually a data base containing generic descriptions of objects and situations, available imagery, and techniques, in addition to topographical and cultural information found in conventional maps.

We recognize that within the limitations of the current state of image understanding it is not possible to replace a skilled photo interpreter. It is possible, however, to greatly facilitate his work by providing a number of collaborative aids that relieve him of his more mundane and tedious chores [1].

II OVERVIEW OF HAWKEYE

Our work has been centered on evolutionary development toward an integrated interactive system. The system consists of an interactive display console, a map data base, an image library, general image analysis routines, and task specialist routines. The capabilities described here have been demonstrated as independent programs that share only data files. We are in the process of integrating them into a single coherent system, known as Hawkeye. Users communicate with Hawkeye naturally, in free-form English and via interactive graphics. The following scenario illustrates the major capabilities that have been demonstrated to date.

The first task when a new image enters the system is to establish correspondence with the map. This is accomplished automatically, by selecting potentially visible landmarks (using navigational data associated with the image) and then locating them in the image using scene analysis techniques. The next step is to confirm the validity of existing knowledge. The system can automatically verify the presence of certain cartographic features, such as roads and waterways, and can also monitor the status of some typical dynamic situations, such as ships berthed in harbor or boxcars stored in a classification yard. New features are identified and incorporated into the data base through the use of a number of interactive aids for mensuration and tracing. For example, new roads can be traced, or heights of bridge supports can be measured.

The system can now use the data base to answer simple queries, such as "show me Pier14", "what is this building?" or "how high is that mountain?". These queries are entered by a photo interpreter via keyboard and display cursor. It also has the potential for responding to a more complex query, such as "how many ships were in Oakland-Harbor yesterday?", by retrieving the relevant image from the library, and then invoking the appropriate task specialist. The system has the ability to accept such requests entered remotely (say, by intelligence analysts) and execute them automatically if it understands, or else relay them to the user (the photo interpreter) for interactive execution.

At this time, the questions that can be handled automatically are limited by the present small size of the data base and the available specialist routines, which automate tasks carefully chosen to exploit existing primitive low-level vision capabilities. Demonstrated capabilities do, however, show the potential of bringing image understanding and artificial intelligence approaches to bear on problems in cartography and photo interpretation.

III TECHNICAL DETAILS

* Visiting from the Department of Computer Science, University of Rochester.

A Image Analysis

In this section, we describe the components of Hawkeye that are directly concerned with image analysis functions, such as those illustrated in the scenario.

1. Map/image correspondence

The first task in the scenario is putting the sensed image into geometric correspondence with reference imagery or a map data base. This is fundamental to virtually every military application of imagery. Our initial approach was a modest improvement on conventional image correlation. Given an image, such as Figure 1, and approximate viewpoint, the system determined potentially visible landmarks and then retrieved from the library images containing the landmarks. Figure 2 shows a selected reference image with the area of overlap and the contained landmarks overlaid on it.

For each landmark, an appropriate area of the reference image was extracted and reprojected to make it appear more similar to the sensed image. The reprojection was accomplished using a camera model, calibration data associated with the reference image, and elevation data obtained from the map. The reference image fragment was first projected down onto the ground plane, and thence back up onto the image plane of the sensing camera. Each reprojected image fragment was then correlated in a small predicted area of the sensed image, using Moravec's high-speed algorithm [5]. Figure 3 shows details of the sensed (right top) and reference (left top) images near a landmark. The bottom left detail is the 16x16 image chip surrounding the landmark automatically extracted for use by the system. The landmark is sought in the area delimited by the large square in the sensed image, and the best matching area is shown at bottom midright. The reprojected version of the chip is shown at bottom midleft, and the best matching area at bottom right. Note that the reprojected reference image more closely resembles the sensed image and that the point of correspondence is therefore more precisely located. Figure 4 illustrates improved reliability: without reprojection, the best match is at the wrong location (indicated by X).

The matching process is repeated for all landmarks expected to be visible. This yields a set of points in the sensed image, with each point corresponding to a particular landmark (Figure 5). From the pairs of corresponding image and world locations, the exact camera parameters for the sensed image were computed by solving an overconstrained set of equations. We can determine a least-squared-error solution either directly analytically, or by an iterative parameter optimization process: the latter has the advantage that any known constraints on parameter values can be readily imposed.

The reprojection technique (unlike currently used techniques) permits the use of reference images that differ radically in viewpoint from the sensed image. Even an oblique image, such as shown in Figure 6, can be matched against the same reference image. Figure 7 shows matching for a single landmark. The views are so different that a meaningful match is impossible without reprojection.

Although reprojection prior to matching is an improvement on conventional image correlation, the fundamental limitation of the correlation approach, namely sensitivity to viewing conditions, remains. In particular, it still cannot match images obtained from radically different viewpoints when the three-dimensional scene structure is complex, from different sensors, or under different illumination or climatic conditions; and it cannot match images against symbolic maps. To overcome these limitations, we developed a new approach, which we call parametric correspondence, for matching images directly to a three-dimensional symbolic reference map.

The map contains a compact three-dimensional representation of the shape of major landmarks, such as coastlines, buildings, and roads. An analytic camera model is used to predict the location and appearance of landmarks in the image, generating a projection for an assumed viewpoint. Correspondence is achieved by adjusting the parameters of the camera model until the predicted appearances of the landmarks optimally match a symbolic description extracted from the image. The matching of image and map features is performed rapidly by a new technique, called "chamfer matching", that compares the shapes of two collections of shape fragments, at a cost proportional to linear dimension, rather than area. These two new techniques permit the matching of spatially extensive features on the basis of shape, which reduces the risk of ambiguous matches and the dependence on viewing conditions inherent in the conventional correlation based approach. The techniques are described in more detail in [4] and [3]. They have obvious application to navigation as well as photo interpretation.

Having placed the image into parametric correspondence with the three-dimensional map, we are now in a position to predict the image coordinates of any feature in the map. Figure 8 shows two pictures with the same section of coastline from the map superimposed on each. This facility is used in monitoring to indicate exactly where in the picture to look. Conversely, we can predict the map features corresponding to any point in the image. This can be used to facilitate interactive graphical communication between the photo interpreter and the data base. In Figure 9, the user has two images displayed simultaneously and can point with a cursor at a location in one image and have the system indicate the corresponding point in the other. (To perform the latter function accurately, the system needs to know the three-dimensional nature of the terrain. We are still in the process of setting up terrain data in the map data base, so in these examples the user supplied the fact that the area in question has roughly constant elevation.)

Using the camera model and image calibration permits many photo interpretation mensuration tasks to be accomplished simply. Routines exist for determining location, length, height, or straight-line distance for features indicated interactively in the image. In Figure 10, the user is measuring the height of a bridge support. Velocity of objects (e.g. ships or cars) indicated in two images can also be determined. In Figure 11, the user indicated a ship in one image, and the system used the landmark finding process to locate the same ship in the other image and hence to determine speed from the deduced distance and the known time delay between the pictures.

The camera model provides a unifying theoretical foundation that subsumes what would otherwise be a collection of ad hoc trigonometric techniques [6]. Combining the map and calibrated image, the system can also, for example, determine alternative routes and travel distances along roads between indicated points.

2. Map-guided monitoring

Having a map and image in correspondence makes many monitoring tasks simpler, because the map can indicate where to look and what to look for in the image. It is important, however, to keep in mind that a map is only an approximation to reality: it may be incomplete, be out of date, suppress details, or contain errors. In order to monitor or to make a detailed interpretation of an image, it is necessary to locate image coordinates of objects more precisely than can be predicted using the map and calibration. In other words, we need routines which can take predictions and verify them in the image. As a first step in that direction, we developed a guided line tracing routine that accepts a rough approximation to the path of linear features, such as rivers or roads, and extracts a best estimate of the precise path in the image. It operates by applying a specially developed line detector in the vicinity of the approximate path and then finding a globally optimal path based on the local feature values [2]. Figure 12 shows the predicted course of a road in a rural area (darker line). The same road has also been predicted without making use of the elevation information in the map (lighter line): note that this prediction is considerably in error. Figure 13 shows the result of the tracing process, obtained fully automatically.

The tracing routine can be used in two ways: to verify the presence of known cartographic features, using prediction from the map and to interactively trace new features for incorporation into the map, using a guideline sketched by the user. The tracing of linear features is currently a tedious manual process that constitutes a major bottleneck in map production [1] [7].

Having a map and image in correspondence makes the automation of many monitoring tasks feasible. Keeping track of boxcars in a railyard, for example, is a typical tedious photo interpretation task. Knowing the layout of the tracks, makes the task essentially a one-dimensional template matching problem. A routine has been developed which flies statistical operators along a track line to hypothesize possible ends of boxcars. These hypotheses are used with knowledge of standard boxcar lengths and characteristics of empty track to locate the gaps between boxcars. The program then marks the cars with a red dot Figure 14 and reports their number, classified by length [2].

Estimating highway traffic is a problem of significant military importance, which could be approached by flying car and truck templates [8] along the path determined by the guided road tracer. We plan to attack this problem in the near future (see Section IV).

Monitoring the presence of ships in a harbor is particularly easy to automate when the map contains details of berths. Given a question about the status of a particular harbor at a particular time, the appropriate image is retrieved from the data base. The ship monitoring routine then projects berth locations from the map onto the image (Figure 15) and uses an edge histogram of that region to determine whether the berth is occupied (Figure 16). The same process works equally well for vertical or oblique imagery as shown in Figure 17.

The key to automatic monitoring lies in having the capability to place the image into correspondence with the map, which then accurately specifies where to look. A relatively simple test may then be used in that limited context. We have implemented three representative demonstrations of this approach and believe that many others are possible, especially in remote sensing [9]. In a production environment, such monitoring could be performed automatically on a continuing basis as new imagery arrived.

B. System Integration

In this section we discuss the integration of the above Hawkeye components into a useful photo interpretation system.

1. System organization

The Hawkeye system consists of several independent processes (forks) which interact by means of inter-process messages. Each process performs a specific set of functions, either for the use of other Hawkeye processes (e.g. the display handler) or in the direct interests of the human user (e.g. a top-level task specialist, such as the railyard monitor). The former processes are "server processes", whereas the latter may be classified as "user processes."

A server process is associated with each external connection, that is, device or data base. Each server presents a standard interface to the rest of the system. Thus, knowledge of the idiosyncrasies of a particular device or data base is required only within the process dedicated to it.

The Hawkeye system currently consists of the following basic modules :-

- Natural language interface
  - help facility
  - command interpreter
  - question answering

- Graphics tablet
  - digitization of pictorial data (maps, photos)
  - graphical communication (pointing, menus)

- Display
  - shared access to display by multiple modules
  - graphical communication

- Generalized map knowledge base
  - repository of cartographic and cultural data
  - generic definitions of semantic objects
  - image library and index
  - question answering about data base

- Map/image correspondence
  - determination of camera and digitization parameters
  - determination of transformations between map and image

- Task specialists
  - mensuration
  - road tracer
  - railyard monitoring
  - harbor monitoring

Each module is written in an appropriate language (INTERLISP, SAIL, FORTRAN, or MACRO) with its own data structures. The total size of these programs exceeded the address space of the host machine (KL-10) several times over.

The user communicates with the system via the natural language interface module (written in INTERLISP) which then calls upon appropriate "server" modules to carry out his request. The user interface module is also responsible for tasking and setting up the system's server modules.

The map/image correspondence and task specialist modules were discussed above, under the heading of Image Analysis. The following sections describe the supporting modules and the details of inter-module communication.

2. Inter-module communication:

Inter-process communication is implemented using the Inter-Process Communication Facility (IPCF) of the TOPS-20 operating system. This facility, which only recently became available, provides significantly better interaction than the pseudo-teletype and shared-page facilities to which we were limited under TENEX. IPCF enables processes (including forks and jobs) to send and receive messages in the form of packets, up to a page (512 words) in length. Messages are copied from the sender's address space and placed in an input queue in system space. The packet remains in the queue until the receiver requests it, at which time it is copied to the receiver's address space.

Messages consist primarily of requests and responses with the format:

Source ID : Destination ID :
Message ID : Message Data

The source and destination ID's are the unique process ID's assigned to the associated processes at the time they were created. The message ID identifies the current request.

Once a request is posted, the requester (sender) will usually wait until the requestee (receiver) has finished processing the request. The requestee, on the other hand, is typically a server and operates by waiting for a request, processing it to completion, sending a response, waiting for the next request, etc. Servers are not interrupted by incoming requests while processing a previous request. Thus, we do not, in general, have a coroutine structure. Such a scenario arises from the fact that processes are necessary not so much to achieve parallelism, but to acquire a larger address space and to integrate multiple languages.

Interface routines are needed to match the basic message passing machinery to the various programming languages used in implementing Hawkeye modules. For example, LISP interfacing routines will accept arbitrary objects (lists, arrays, numbers etc.) for transmission: One such routine accepts any LISP S-expression, and transmits it to another LISP process that evaluates it and returns the result.

3. Display server

The display server in a multi-process system has responsibility for allocating portions of the display area to requesting processes. For simplicity and modularity each process should believe that it has access to its own private display for presenting pictures and graphics, and for obtaining graphic input from a cursor. In actuality, processes are allocated windows (regions of the physical display area) and the display server performs the necessary coordinate transformations.

4. Graphics tablet server

The tablet server has a similar role to that of the display server, except that it is used only for graphical input. The surface area of the digitizing table may contain many different documents, such as a conventional topographic map, photographs of the area, and command menus. The tablet can be used in many ways. Landmarks in maps or photos can be manually indicated, and thus the correspondence between the document and the world model established. Features can then be traced for input to the map data base, using routines which display the tracings and permit real time editing. The tablet cursor can be used just like the display cursor for pointing in mensuration and question answering tasks.

5. Map data base server

The data base server is the means of access to the map data base for other system modules which are not required to have detailed knowledge of its structure or implementation. This server contains access routines for answering a variety of standard queries about specific data and the general format of the data base: for example, "what is at (x,y,z)", "what is the closest road to (x,y,z)", "what roads are contained in the area bounded by ...", "where is Oakland Mole", "what is the <attribute> of <object>", "what attributes does <object> have", and so forth.

The map data base contains three-dimensional descriptions of cartographic and cultural features, including coastlines, major roads, lakes, bridges, airfield runways, oil storage tanks, and harbor lights. In addition, the map contains a partial taxonomy of world entities, with relevant general semantics, information about available imagery, and descriptions of data structures used by the system. The information about imagery includes file name, calibration data, and geographic area covered and can be used in selecting appropriate pictures for specific tasks. The descriptions of the data structures enable the system to construct automatically new entities of the correct structure for inclusion in the data base.

The map data base is a disk-based semantic net data structure that can contain realistic quantities of data represented in a way which permits efficient access. Entities are represented by LISP atoms (e.g. English words), and information associated with the entity is stored in a property list format. Relationships to other entities are also stored on the property lists, thus establishing a network structure in the data base. When information concerning a particular entity is sought, the property list is retrieved from disk and established in core. A "paging" scheme limits the amount of data in core (to, say, 1000 entities) and writes entities back out to disk, if necessary, the least recently used ones first [2]. Retrieval of the information is by means of a hash table on disk, which means that access time is constant and independent of data base size. The geometric data are indexed (the index structure is part of the data base) via K-D trees [10], one tree for each class of entity sought, to enable fast retrieval of information relevant to a particular area. We are continually refining representations for the basic map entities in order to increase the richness of information and retain efficiency of retrieval.

We are setting up a map of the San Francisco Bay Area, containing major features, coastlines, bridges, and highways. Figure 18 is a portion of a U.S. Geological Survey (USGS) map of the area; Figure 19 shows the portion of the map currently in the data base. Figure 20 shows part of the map at higher resolution. The map consists of about 4000 points, plus various semantic relationships, totaling about three-quarters of a million bytes of disk storage. (Access to a particular item of information takes less than a millisecond if it is paged in, and fifteen to thirty milliseconds plus disc access time if it has to be read in). The map information is entered by manually tracing features on a USGS map using a digitizing table: map data in digital form are not available, and the problem of digitizing printed maps has rather different constraints from the problem of making maps from photographs, so we could not exploit our guided tracing techniques. We will soon bring up a terrain data base which will provide the elevation of any location in the map.

6. User interface

When a system becomes large and complex, ease of user interaction is essential. The user interface module provides natural language communication with Hawkeye. Capabilities include querying the data base, commanding actions, such as calibration of an image, mensuration, or monitoring, and querying the system about available facilities and how to use them.

The user interface is implemented with LIFER, a proprietary language definition and parsing system developed at SRI by Hendrix [11]. LIFER uses an augmented transition net grammar whose symbols correspond to semantic as well as syntactic entities. LIFER makes it particularly easy to achieve robust dialogs about a limited domain, facilitated by such features as acceptance of elliptical input and the ability to expand the grammar incrementally in English as deficiencies are discovered.

LIFER interfaces have been designed for several large AI programs including the ACCAT test-bed supported by ARPA [12]. A unique requirement arising with pictorial data is the need for graphical communication, such as pointing with a cursor in an image, in conjunction with natural language commands. For example, "What is this ?" or "What is the distance between here and here ?". The LIFER grammar has been written to parse such expressions, requesting coordinates from the servers providing graphical input.

The natural language interface permits tasking via requests in English. Hawkeye will notify the user when requests arrive, printing them if he desires. The user can ask the system to read the requests, parse and execute them if it can. The system will alert the user if it cannot understand or carry out a request, so that it may be fulfilled interactively. It will also notify him when it has finished all outstanding tasks.

IV NEW DIRECTIONS: ROAD MONITORING

Hawkeye demonstrated the feasibility of using knowledge about maps and imaging to automate a variety of representative photo interpretation tasks. With this knowledge, adequate performance was achieved in straightforward cases, but the system was easily misled by contingencies that it did not know about, for example, clouds. In order to approach human performance, substantially more world knowledge is necessary. In the next stage of our research, we plan to develop a system with considerable expertise in a specialized task area.

The task we have selected is that of monitoring traffic on roads. More specifically, given a sequence of reconnaissance images of a region under surveillance, possibly taken under adverse viewing conditions, the system will first locate sections of known roads visible in the images, locate anomalous regions on the roads whose size, shape, velocity and other characteristics are consistent with those of vehicles, and then perform a detailed scene analysis in the vicinity of the anomalies in order to identify specific vehicle types.

The road monitoring system is being organized as two expert subsystems designated the "Road Expert" and the "Vehicle Expert". The Road Expert will have the task of analysing imagery at low resolution to:

- Establish image map correspondance (Camera Calibration Algorithm).
- Locate in the imagery the visible segments of roads to be monitored (Low Resolution Road Detector).
- Accurately mark the road boundaries and determine their map coordinates (Intermediate Resolution Road Detector).
- Search for anomalies within the marked road boundaries (Road Anomaly Detector).
- Confirm potential vehicle existance by comparison with previously acquired imagery, and with general knowledge of vehicle behavior using data base support (Symbolic Reasoning Module).

The "Vehicle Expert" will examine at high resolution the potentially interesting objects found by the road expert. It will have the task of:

- Producing a description of the 3-dimensional geometry of the road objects. (Geometric Description Module).
- Comparison of observed object descriptions with stored descriptions of vehicle types (Vehicle Identification Module).

In order to attain the level of performance for which we aim, the system will require knowledge of a wide variety of situations and events, such as obscuration of roads by trees or clouds, the visual effects of snow and rain, the behavior of roads at intersections, mountains, tunnels and so forth. The system will also require knowledge of its repertoire of resources, their abilities and limitations, and how to evaluate its own performance. The Hawkeye system framework provides a suitable foundation for integrating all the capabilities and knowledge into a unified system.

V ACKNOWLEDGEMENT

The research was supported by the Defense Advanced Research Projects Agency of the Department of Defense and was monitored by the U.S. Army Research Office under Contract No. DAAG29-76-C-0057.

REFERENCES

1. Barrow, H. G., et al., "Interactive Aids for Cartography and Photo Interpretation", in Artificial Intelligence--Research and Applications, Annual Progress Report to ARPA, Contract DAAG29-76-C-0012, Artificial Intelllligence Center, Stanford Research Institute, Menlo Park, CA 94025 (June 1976).

2. Barrow, H. G., "Interactive Aids for Cartography and Photo Interpretation", Semiannual Technical Report to ARPA, Contract DAAG29-76-C-0057, Artificial Intellgence Center, Stanford Research Institute, Menlo Park, CA 94025 (November 1976).

3. Barrow, H. G., "Interactive Aids for Cartography and Photo Interpretation", Semiannual Technical Report to ARPA, Contract DAAG29-76-C-0057, Artificial Intellgence Center, Stanford Research Institute, Menlo Park, CA 94025 (May 1977).

4. Barrow, H. G., Tenenbaum, J. M., Bolles, R. C., Wolf, H. C., "Parametric Correspondence and Chamfer Matching: Two New Techniques for Image Matching", In *Proceedings of the ARPA Image Understanding Workshop*, Minneapolis, (April 1977).

5. Moravec, H. P., "Techniques Towards Automatic Visual Obstacle Avoidance", submitted to 5IJCAI.

6. "The Manual of Photogrammetry", Ed. R. N. Colwell, published by American Society of Photogrammetry, Falls Church, Virginia (1972).

7. Garvey, T. D., and J. M. Tenenbaum, "Application of Interactive Scene Analysis Techniques to Cartography", Tech Note No. 127, Artificial Intelligence Center, Stanford Research Institute, Menlo Park, CA 94025 (November 1976)(also published in Proc. 3IJCAI).

8. Binford, T. O., In Proceedings of the ARPA Image Understanding Workshop, University of Southern California, April 1976.

9. Tenenbaum, J. M., et al., "Research in Interactive Scene Analysis", Final Report to NASA, Contract No. NASW-2865, Artificial Intelligence Center, Stanford Research Institute, Menlo Park, CA 94025 (December 1976).

10. Bentley, J. L., "Multidimensional Binary Search Trees Used for Associative Searching", CACM, Vol. 18, No. 9 (September 1975).

11. Hendrix, G. G., "The LIFER Manual: A Guide to Building Natural Language Interfaces," Tech. note 138, Artificial Intelligence Center, SRI International, Menlo Park, CA, February 1977.

12. Sacerdoti, E. D., "Language Access to Distributed Data with Error Recovery", Proceedings Fifth IJCAI, Cambridge, Mass., August 1977.

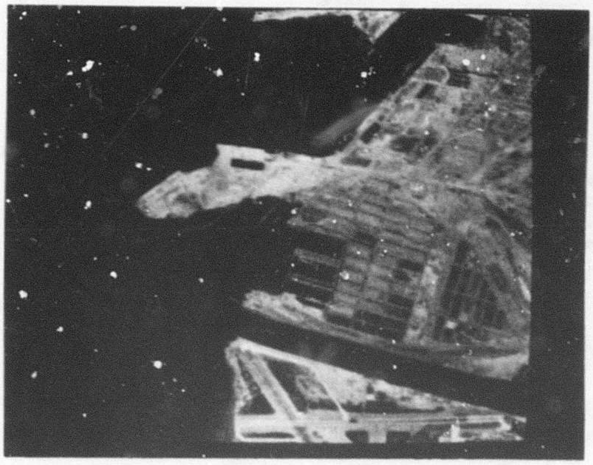


Figure 1. A new sensed image

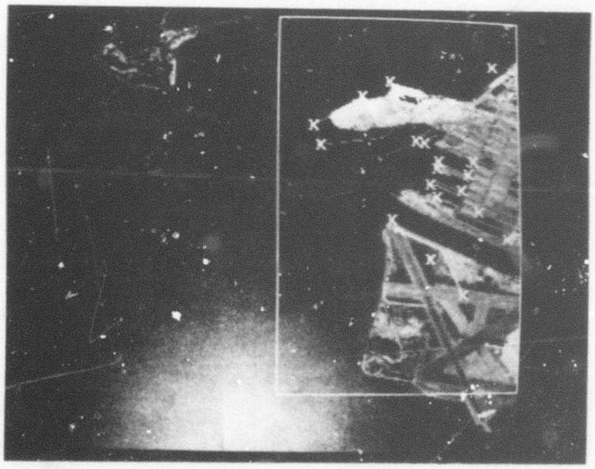


Figure 2. A selected reference image and landmarks

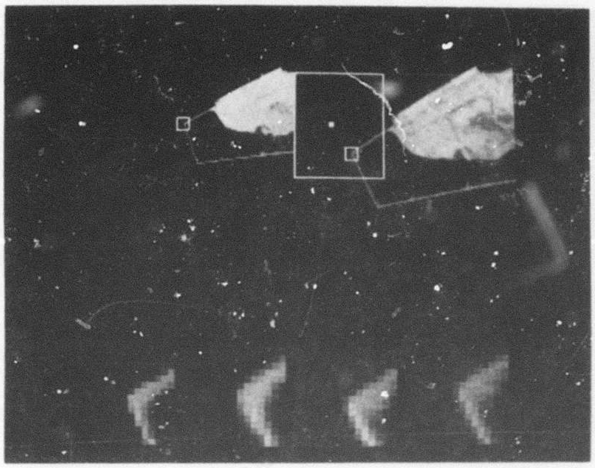


Figure 3. Correlation matching of an image chip

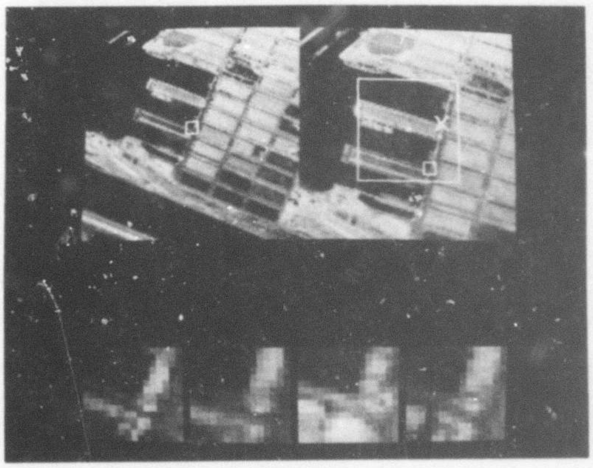


Figure 4. A mismatch with an unreprojected chip

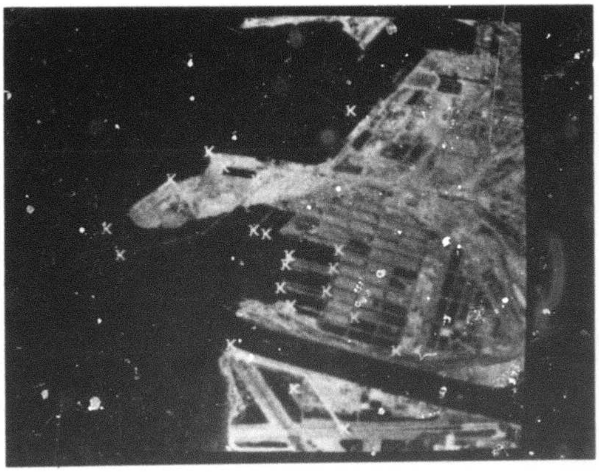


Figure 5. Landmarks located in the sensed image

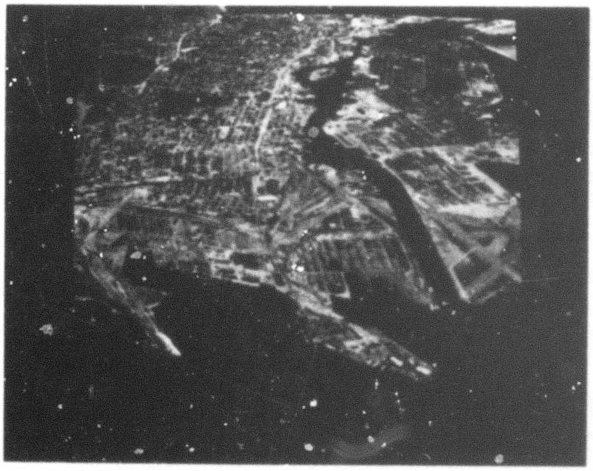


Figure 6. An oblique sensed image

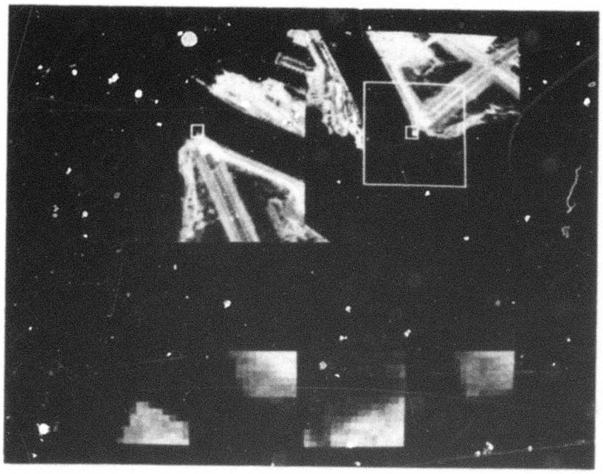


Figure 7. Matching vertical and oblique views

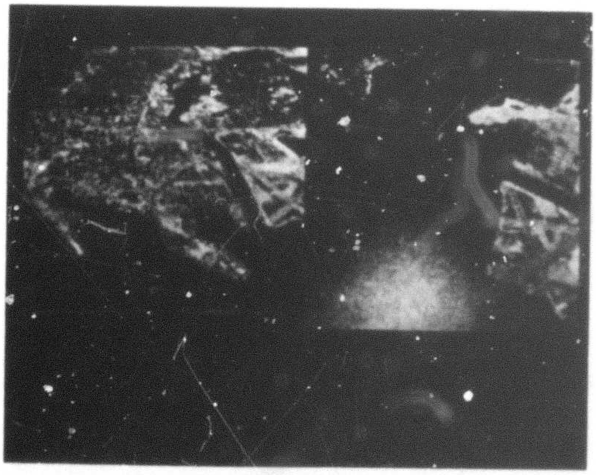


Figure 8. The map projected onto two pictures

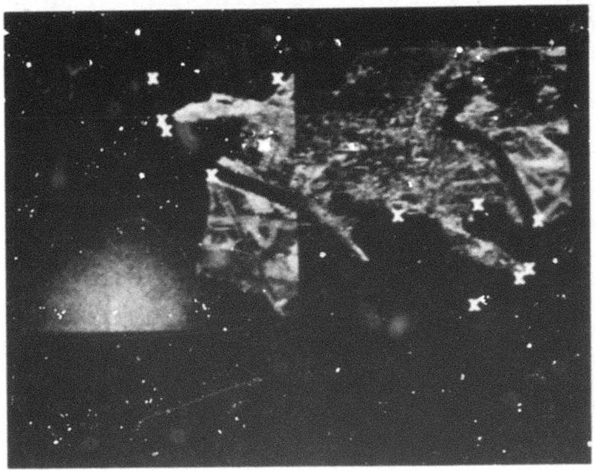


Figure 9. Indicating corresponding points

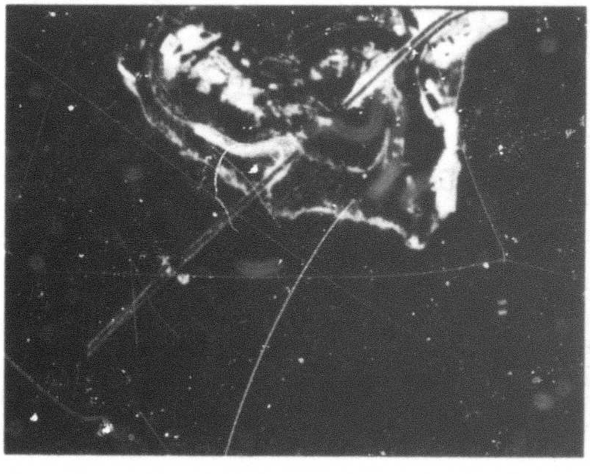


Figure 10. Measuring the height of a bridge

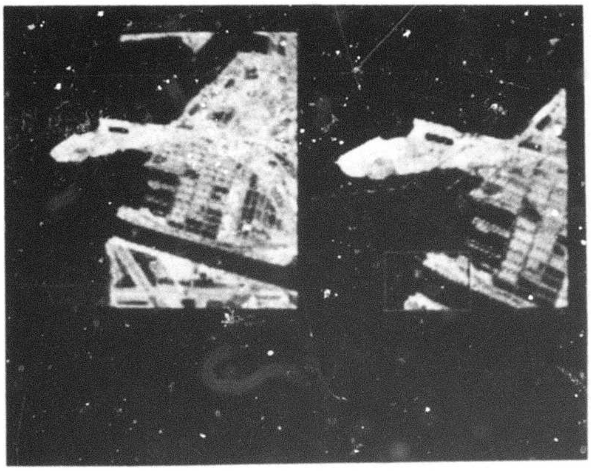


Figure 11. Measuring the speed of a ship

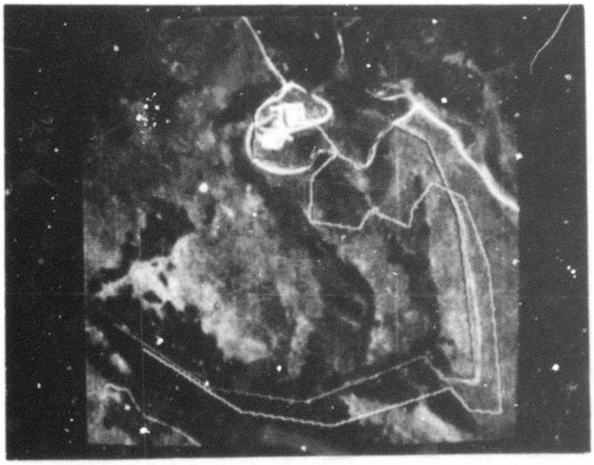


Figure 12. A road predicted from the map

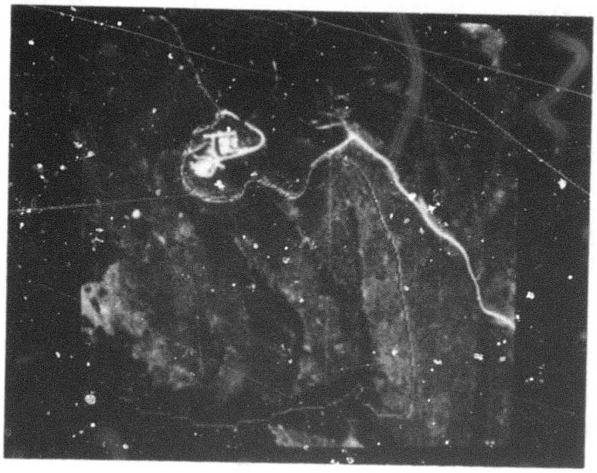


Figure 13. The road after automatic tracing

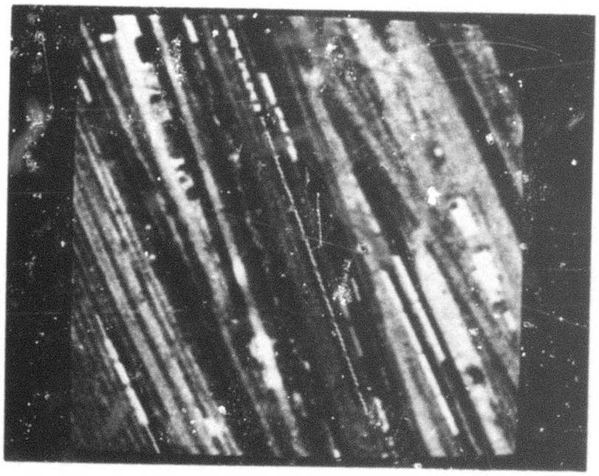


Figure 14. Boxcars counted automatically

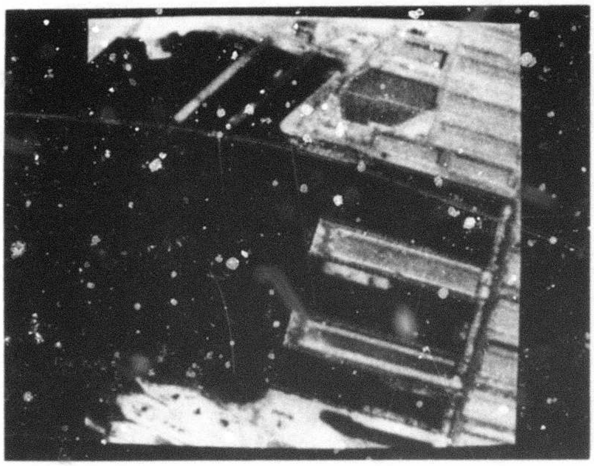


Figure 15. Harbor piers predicted from the map

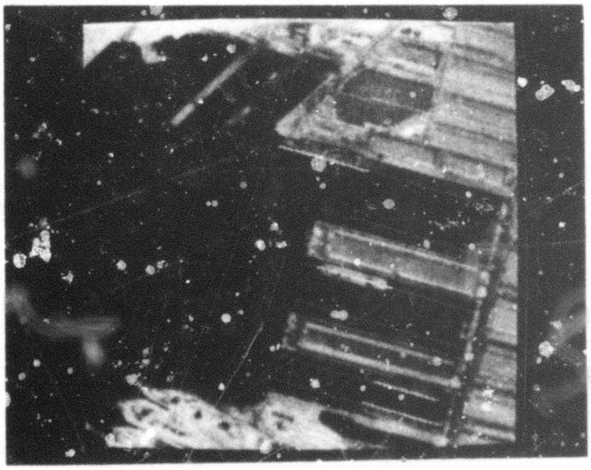


Figure 16. Berthed ships detectec

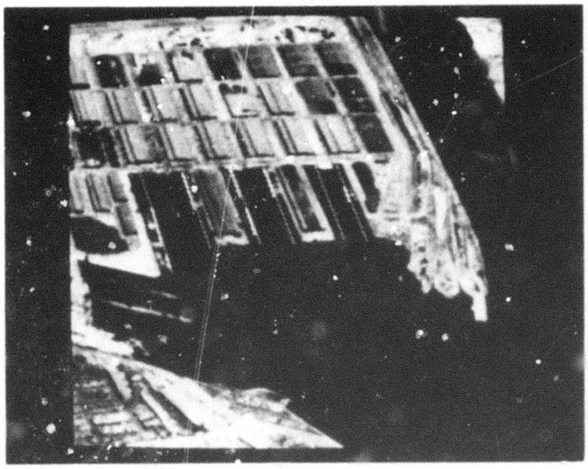


Figure 17. Ber'hed ships in an oblique image

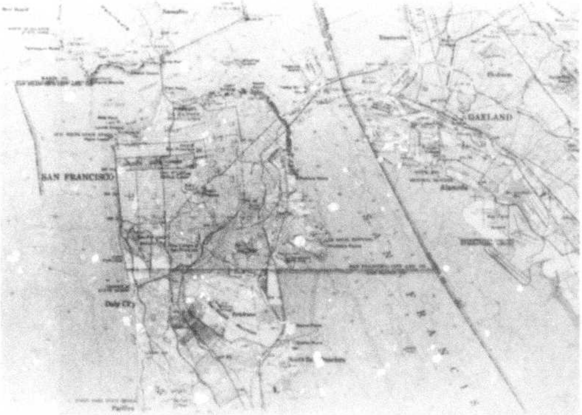


Figure 18. A USGS map of San Francisco Bay

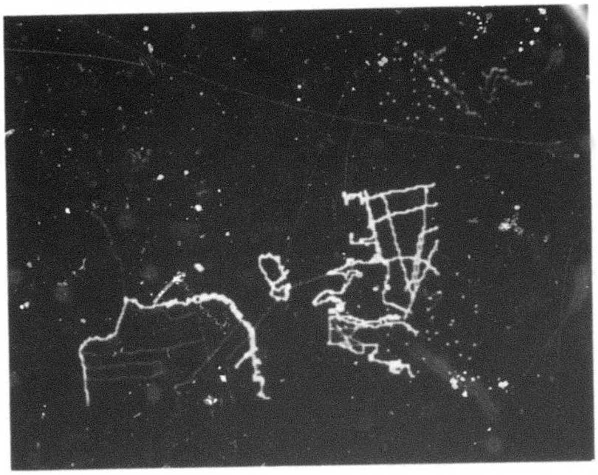


Figure 19. Display of the digital map data base

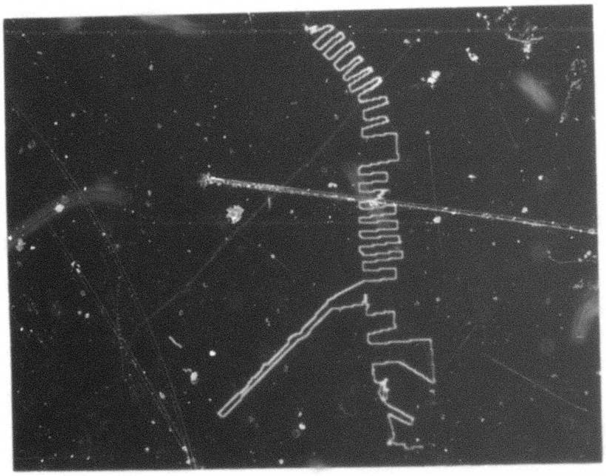


Figure 20. The map at higher resolution

Spatial Understanding Overview

T.O.Binford

Stanford University

The objectives of this research are to interpret image sequences in terms of spatial features and spatial relations, and to use shape knowledge in low-level and high-level vision. The research is directed to applications in stereo photointerpretation, stereo change detection, and terminal guidance for strategic and tactical devices. The goal is to develop techniques for passive ranging and interpretation which can be used in guidance and monitoring despite differences in sensors, viewpoint, sun angle and weather or surface conditions such as rain or snow.

Collaboration

We now have a collaborative arrangement with the Signal Processing Laboratory of Lockheed Missiles and Space Co. Support is being sought for a joint proposal involving Stanford University, Lockheed, and SRI. In this arrangement, Lockheed would provide system integration and implementation, along with input concerning military relevance.

Representation

We have begun work on representing commercial aircraft for use in interpretation of airfield imagery. A new program for representing complex objects and displaying symbolic structures in their appearances is being designed.

Depth Mapping

An earlier level program based on obtaining range measurements at a coarse sample of points was described in previous proceedings of the Image Understanding workshops. This sample was dense enough to obtain ground surface descriptions. A program which makes more dense and uniform range mapping is described in detail in the paper by Gennery in these proceedings. Application is shown to an aerial view of a parking lot scene, a ground level parking lot scene, and an aerial view of an apartment building. In all three scenes, it locates the ground surface and finds areas above the ground surface, with some mistakes. In the apartment house scene, it picks out the roof surface also.

In the near future, the depth mapping program will be used to produce maps of the Pittsburgh scene of CMU, and portions of passenger terminals at San Francisco airport. To deal with large images, initially they will be done in smaller pieces. We expect to make a version of the program which proceeds row by row to cut down the amount of storage necessary which will allow processing entire pictures of the size of the Pittsburgh series.

A high resolution range mapping program will follow which determines object boundaries to higher resolution and which can discriminate small regions.

Interactive Vision

A new direction of research has begun whose goal is a Mycin-like system in which image understanding programs can be built by non-expert users. It is intended to generate programs to find airfields and oil tanks, in a way that extends in a natural way to a much larger class of image understanding tasks. The system depends upon a program which draws conclusions from the representation of shape of objects.

Database

We have extended our data base to include a mapping sequence of aerial photographs of San Francisco airport suitable for stereo. We are negotiating for use of another mapping sequence to use for stereo change detection. We have obtained a stereo pair of urban areas in Pittsburgh from Carnegie Mellon University. We have arrangements to get digitized data from a SAR flight, along with photographs which cover the same area. We are grateful to CMU, USC, Lockheed, Hughes and the engineering department of the San Francisco International Airport for valuable assistance in obtaining and digitizing imagery.

OVERVIEW OF THE ROCHESTER IMAGE UNDERSTANDING PROJECT

Jerome A. Feldman

Computer Science Department
The University of Rochester

1. Basic Theory

Our basic approach to vision was set out in [Ballard and Brown, 1977]. The query-driven, top-down approach to vision is especially well suited to application as opposed to purely fundamental, non-applied research.

The achievement here is to carve out a useful subset of ideas and techniques from the many proposed for vision, and to organize them in a flexible and extensible way in the service of particular domains.

Our three-layered structure (image data, sketchmap, and model) and associated constructs allow the use of previously acquired results and the increasing automation of image-understanding decisions. Our applications programs use these basic ideas. In writing the applications programs (involving aerial and biomedical images) we have improved and tested our ideas on particular vision mechanisms (executive procedures, constraint networks, location descriptors, etc. [Ballard, Brown and Feldman, 1977]), data structures, and control of vision processing (see Section 2).

2. Application to Ship Finding

In this simple application of the system, docked ships are located in harbor scenes [Brown and Lantz, 1977]. The system, under direction of the user-written query, begins by deciding where to look by satisfying a constraint network; the more information provided, the narrower the focus of attention. Recent work at SRI [Barrow, 1977] has shown that map data may be automatically registered with images such as ours to within better than a pixel, so we felt comfortable about bypassing the registration problem in this study. Were the registration uncertain, the constraints would produce a more fuzzy area to search than they did.

This application has provided a test bed for the constraint network idea, as well as for the representation of subsets of 2-space (linear and area objects) and for set-theoretic operations on them (union and intersection).

3. Display of 3-D Data

In many applications it is useful to characterize the orientational properties of 3-D structures. For example, the surface normals of mountainous areas are more varied in direction than those of plains areas. Two descriptions of 3-D orientation for general 3-D wire-frame structures were designed [Brown, 1976a; Brown, 1976b; Brown, 1977]. The descriptions provide a generally useful tool for display of certain 3-D volumes, especially well suited for raster graphics [Brown and Selfridge, in preparation]. Some basic mathematical/statistical questions raised here were answered by J. Wellner in our Statistics Department [Wellner, to appear], and he and his students are presently applying the theoretical results to provide practical statistical tests for 3-D vector data [Wellner, in preparation].

4. Component Building

4.1 Hardware

A second Eclipse computer was purchased for use by the Vision Laboratory. It has its own ETHERNET board (also acquired), and will control a Grinnell high-resolution color raster display device (also acquired, but not yet in house). For all this, and to free up the RIG system, new memory boards were purchased and their controller designed, built, and debugged. Acquisition of an image input device is proceeding; upgrading low-latency mass storage for images is also proceeding. All of the above were acquired with non-DARPA funds. The disk unit budgeted for this period has been ordered, but has not yet arrived.

4.2 Software

Basic software support for the system has been operational for six months and is improving daily (see Section 7). Communications programs and protocols are under development for distributed computing. Controlling programs for the Grinnell display are being written. SAIL code has been written for the basic data structures in the vision system and for operations on them. SAIL code has been obtained from Carnegie-Mellon University (their entire vision laboratory package), Stanford, and USC (e.g., the Heuckel operator), and made to run on our system. Many

SAIL utilities for image management, transformation, and transmission have been written. A general Header for use in transmission of images in distributed computing environments has been designed and is in use. An image protocol for use in distributed computing environments has been designed [Maleson, Rashid and Nabielsky, 1977], to allow interprocess communication of image data (see Section 8).

5. Texture Understanding

Standard texture analysis techniques rely on the calculation of a set of features (like edge probability per unit area, or local neighborhood co-occurrence probability matrices) on training sets of images, taking statistical measures of these features for each training set (mean, standard deviation, entropy, etc.), and partitioning the feature hyper-space so that each partition contains exactly one training set. Unknown texture patches are now measured by the same feature operators to determine their location in feature hyper-space, and are assigned the texture class of the appropriate partition. This technique works well for limited domains, where an accurate training set can be chosen, and where textures exhibit variation in the local features measured. Rotations and scale changes result in a new texture class assignment.

We are approaching the texture problem by dividing texture regions into meaningful subelements of similar intensity sample points, then using rotation- and scale-invariant shape measures to characterize these regions, and finally determining spatial relationships among our subelements. By using a decision-tree program structure, easily discriminated textures are separated quickly, and more complex textural structure is only extracted when necessary. This texture analysis scheme not only classifies texture patches into sets, but also produces a description of similarities and differences among different patches. That information is then available to higher-level semantically-driven processes, and is more useful than a binary same/different decision. In this period, we have completed a prototype texture analysis system that demonstrates the feasibility of this approach [Maleson, 1977].

6. Semantic Nets, Frames, and Associations

A knowledge representation system was developed which is based on the use of a semantic net on which a higher-level structure of frames has been superimposed. The system was designed for use with a natural language system which is especially concerned with finding the correct senses of ambiguous words in context. An examination of several linguistic examples shows how the representation system facilitates associative searches of context for potentially appropriate senses of ambiguous words, and how the results of such searches can often provide definite referents [Hayes, 1976; Hayes, 1977a; Hayes, 1977b]. The applications of this model to the image understanding world modelling problem are being explored.

7. Support System Development

The major accomplishment of this period was the bringing of the RIG system into full production use. The RIG system consists of 4 64KW minicomputers (intended primarily for stand-alone use and possessing local disk storage and high resolution raster displays) connected in a 3 MHz ring network to a Data General Eclipse. The Eclipse maintains a modestly large local file capacity (~100 MB), hard copy printing and plotting, and magnetic tape. It also provides editing and other facilities to a number of local terminals, and serves as a gateway between larger campus machines (360/65 and KL 10), the ARPANET (as a VDH), and our local network. We have also obtained funding from the National Science Foundation for an Image Understanding Laboratory, and will add it to RIG in the coming year [Ball et al., 1976; Feldman & Rashid, '77].

8. Image Protocol Development

The Rochester Image Protocol is being developed within the RIG framework and governs communication between image handling processes in our network. It is built around the concept of a structured image definition similar in spirit to the structured graphics display files of [Sproull and Thomas, 1974]. This image data structure serves both as a common language for describing images and as a uniform way of specifying the display of image data on various raster devices (e.g., plotting devices, black and white and color variable intensity and simple intensity displays).

9. Programming Language Development

We developed and made available to the community important corrections and improvements to the compiler for SAIL, the most widely used language in the DARPA image understanding effort. We are continuing to maintain and improve the SAIL language [Rashid, '76] while working on a new system.

Some fundamental properties of distributed computing (DC) do not occur in conventional programming and these properties lead in a natural way to programming language constructs. The most obvious property is that a distributed computation is spread among several computers which are assumed to be connected by some communication paths. For the forseeable future, these communication paths will be less reliable and have lower bandwidth than is available in the processors themselves. This leads us to expect that DC programs will be made up largely of self-contained modules which will share very little information directly. One would also want to have the communication between modules be some asynchronous message protocol rather than subroutine or coroutine calls where one module would always have to wait for a response from the other. It appears to us that the module-message paradigm is inherently well suited to DC and is likely to appear in some form in any proposed high-level language for DC.

Starting from the basic module-message paradigm, we have been attempting to develop a new generation of high-level programming languages which would incorporate as much as possible of the computer science of the last decade. This overly ambitious project is called PLITS (Programming Language in the Sky). In addition to DC, the PLITS effort is attempting to encompass our current knowledge of software reliability, language extensibility, and automatic programming. Current efforts include the construction and use of an interim PLITS and a number of basic studies on particular issues. The most advanced is a careful definition of the PLITS-DC proposals, expressed as a gedanken extension to PASCAL. Other issues are addressed in [Feldman, 1976] and forthcoming papers.

REFERENCES

Ball, E., Feldman, J., Low, J., Rashid, R., and Rovner, P., "RIG, Rochester's Intelligent Gateway: System Overview," TR5, Computer Science Department, University of Rochester, April 1976; also appeared in IEEE Transactions on Software Engineering, Vol. SE-2, No. 4, December 1976.

Ballard, D. and Brown, C., "A Query-Directed System for Image Analysis," Vision Workshop University of Massachusetts, June 1977.

Ballard, D., Brown, C., and Feldman, J., "An Approach to Knowledge-Directed Image Analysis," to be presented at the 5th International Joint Conference in August 1977.

Barrow, H., "Interactive Aids for Cartography and Photo Interpretation," Semiannual Technical Report, SRI, May 1977.

Brown, C., "Neuron Orientation: A Computer Application," in Computer Analysis of Neuronal Structure, R.D. Lindsay (Ed.), Plenum Press, 1976(a).

Brown, C., "Representing the Orientation of Dendritic Fields with Geodesic Tesselations," TR13, Computer Science Department, University of Rochester, September 1976(b).

Brown, C., "Two Descriptions and a Two-Sample Test for 3-D Vector Data," submitted to Technometrics 1977.

Brown, C. and Lantz, K., "Representation and Use of Knowledge in a Goal-Directed Vision System," DARPA Image Understanding Workshop, 1977.

Brown, C., and Selfridge, P., "Raster Graphics Display of Spherical Polyhedra," in preparation.

Feldman, J.A., "A Programming Methodology for Distributed Computing (Among Other Things)," TR9, Computer Science Department, University of Rochester, September 1976.

Feldman, J. and Rashid, R., "System Support for a Distributed Image Understanding Program," DARPA Image Understanding Workshop, April 1977.

Hayes, P.J., "A Process to Implement Some Word Sense Disambiguations," TR6, Computer Science Department, University of Rochester, March 1976.

Hayes, P.J., "On Semantic Nets, Frames and Associations," TR19, Computer Science Department, University of Rochester, August 1977(a).

Hayes, P.J., "Some Association-Based Techniques for Lexical Disambiguation by Machine," unpublished doctoral dissertation, Ecole Polytechnique Federale de Lausanne, 1977(b).

Maleson, J.T., "Understanding Texture in Natural Images," Ph.D. thesis, to appear September 1977.

Maleson, J., Rashid, R., and Nabielsky, J., "The Rochester Image Protocol," Computer Science Department, University of Rochester, Internal Memo, February 1977.

Rashid, R., "Information Retention in Hash Coding: An Associative Data Base," TR8, Computer Science Department, University of Rochester, 1976.

Sproull, R. and Thomas, E., "A Network Graphics Protocol," SIGGRAPH-ACM, Vol. 8, No. 3, 1974.

Wellner J., "Two Sample Sobolev Tests on Compact Riemannian Manifolds," to appear in Statistical Annals.

Wellner, J., in preparation.

IMAGE UNDERSTANDING AND INFORMATION EXTRACTION

T.S. Huang
K.S. Fu

School of Electrical Engineering, Purdue University
West Lafayette, Indiana 47907

OVERVIEW

The objective of our research is to achieve better understanding of image structure and to improve the capability of image processing systems to extract information from imagery and to convey that information in a useful form. The results of this research are expected to provide the basis for technology development relative to military applications of machine extraction of information from aircraft and satellite imagery.

A block diagram of an Image Understanding System is shown in Fig. 1. We first consider the left side of the block diagram. After the sensor collects the image data, the preprocessor may either compress it for storage or transmission or it may attempt to put the data into a form more suitable for analysis. Image segmentation may simply involve locating objects in the image or, for complex scenes, determination of characteristically different regions may be required. Each of the objects or regions is categorized by the classifier which may use either classical decision-theoretic methods or some of the more recently developed syntactic methods. In linguistic terminology, the regions (objects) are primitives, and the classifier finds attributes for these primitives. Finally, the structural analyzer attempts to determine the spatial, spectral, and/or temporal relationships among the classified primitives. The output of the "Structure Analysis" block will be a description (qualitative as well as quantitative) of the original scene. Notice that the various blocks in the system are highly interactive. Usually, in analyzing a scene one has to go back and forth through the system several times.

Past research in image understanding and related areas at both Purdue and elsewhere has indicated that scene analysis can be successful only if we restrict a priori the class of scenes we are analyzing. This is reflected in the right side of the block diagram in Fig. 1. A world model is postulated for the class of scenes at hand. This model is then used to guide each stage of the analyzing system. The results of each processing stage can be used in turn to refine the world model.

Research in image understanding at Purdue concerns with all aspects of the block diagram in Fig. 1. However, the emphasis will lie in the interaction between the processing stages (left side of Fig. 1); and in the searching for suitable types of world models. One type of world model we are looking into combines the ideas of hierarchical relational graphs and the linguistic approach.

SUMMARY OF RESEARCH PROGRESS

We summarize the recent progress of some of our research projects.

(A) A Syntactic Approach to Image Understanding - Janmin Keng and K. S. Fu

A Syntax-Directed Method has been investigated and developed for the land-use classification of satellite images. In particular, the highway, river, bridge, and commercial/industrial recognition have been successfully achieved in a fully automated level. Image segmentation and object detection have also been studied. A syntactic method that utilizes the texture measurements and tree grammar analysis has been devised and tested on different images, such as LANDSAT and aerophotographic images.

(B) Image Segmentation Using Texture and Grey Level -S.G. Carlton and O.R. Mitchell

The research effort underway concerns the application of textural features to the image segmentation problem. The segmentation technique uses a texture measure that counts the number of local extrema in a window centered at each picture point. Four grey level pictures are derived, each of which represents a texture or grey level property of the original image. These intermediate pictures may be viewed as a 4-dimensional image in which each point consists of a 4-dimensional vector. These vectors are then clustered into different groups and averaged to form vectors representative of each group. The segmentation is completed by assigning each pixel in the original image to one of the groups defined by the representative vectors using a distance criterion. This process may be structured hierarchically by repetitively utilizing diminishing window sizes.

(C) Random Field Approach to Contextual Pattern Classification - K.S. Fu and T. S. Yu

We are concerned with the problem of using contextual information for classification of remotely-sensed multispectral data. The problem is interesting because the occurrences of data vectors at resolution elements tend to be correlated. First a torus process is consistently defined on a rectangular torus. The second order moments and spectral density function are specified. The torus process is then extended to form a spatial process defined on the whole infinite plane. The classification algorithm is to employ the Bayesian strategy with the pattern to be classified and its neighbors to provide the optimal decision. Experimental results on LANDSAT data have demonstrated the improvement in classification accuracy when context information is used.

(D) Fourier Descriptors for Extraction of Shape Information - T. Wallace and P. A. Wintz

We have extended the work of Granlund and Persoon and Fu on Fourier descriptors (FD) in several ways. Our present algorithm computes FDs very efficiently using the FFT, the normalization procedure is straightforward and much more efficient than the root-finding technique of Persoon and Fu, and we preserve all of the shape information in the original data. This shape recognition algorithm is powerful and well-tested but it requires detection of the boundaries of objects in photographic data before it can be applied. We have worked with the BLOB boundary-finding algorithm of Gupta and Wintz, modifying BLOB so that it is more effective with ordinary photographic data rather than the multi-spectral data it was originally intended to process. (BLOB locates regions of similar mean and variance in the data.) We recently processed a photograph containing two airplanes with BLOB and then examined the shapes of all contours found with lengths 50 to 1024 with the FD algorithm. The two airplanes were correctly identified by the program, and there were no false classifications. Future plans include extending the FD algorithm to recognize three-dimensional objects, and improving present boundary finding techniques.

(E) Filtering to Remove Cloud Cover in Satellite Imagery - O. R. Mitchell and E. J. Delp, III

We are using homomorphic filtering techniques to remove multiplicative noise effects such as cloud cover and atmospheric turbulence in ERTS imagery. Our present approach is to estimate the cloud statistics directly from the cloudy pictures using a cloud distortion model. Once the noise (cloud) power spectrum is obtained, an optimum filter is derived to separate the signal and noise. The filtering implemented is a three dimensional computerized filter. The three dimensions correspond to two spatial dimensions and one spectral dimension. The distortion model developed for the image includes cloud reflection and transmission and atmospheric turbulence.

(F) Characterization of Context in Imagery by Two-Dimensional Random Processes - P. H. Swain and E. F. Kit

Classification analysis of multispectral image data is routinely carried out by classifying a single pixel at a time, extracting information from the spectral domain, ignoring the two-dimensional or image character of the data. Recent studies confirm that there is useful information in the context of a pixel (e.g., its neighbors) which can be helpful in identifying the pixel. Utilization of context, the information contained in the spatial dependencies among image points, is an important step on the way to achieving "image-understanding". In this research the scene is considered to be a multi-dimensional random process characterizable in terms of its statistical transition properties. Implementation of classification rules utilizing these properties without being prohibitively expensive in terms of computational requirements represents a considerable challenge.

PUBLICATIONS

1. J. Keng, "A Syntactic Method for Image Segmentation," Proceedings of Seventh Annual Symposium of Automatic Imagery Pattern Recognition, Electronic Industrial Association, College Park, Maryland, May 23-24, 1977.

2. T. S. Yu and K. S. Fu, "Contextual Pattern Classification for Remotely Sensed Multispectral Data," Proceedings of the Eighth Modeling and Simulation Conference, Pittsburgh, Pa., April 1977.

3. O. R. Mitchell, C. R. Myers, and W. Boyne, "A Max-Min Measure for Image Texture Analysis," IEEE Trans. on Computers, May 1977.

4. S. G. Carlton and O. R. Mitchell, "Image Segmentation Using Texture and Grey Level," Proc. at the IEEE Computer Society Conference on Pattern Recognition and Image Processing, June 6-8, 1977.

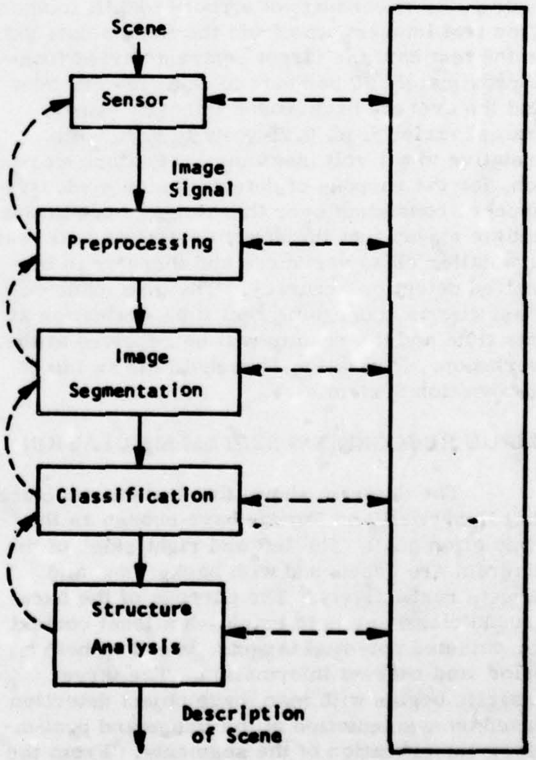


Fig. 1 An Image Understanding System

AUTOMATIC IMAGE RECOGNITION SYSTEM

Program Status, September 1977

R. Larson

HONEYWELL INC.
Systems and Research Center
Minneapolis, Minnesota 55413

This program is entering the second phase, where the effort will be on simulating an airborne tactical recognition system. The Autothreshold hardware development has been completed and we have begun working on the system structure, the data base and some of the component subroutines. This status report is a summary of these initial steps.

AUTOTHRESHOLD HARDWARE

We have tested the Autothreshold ability to adapt to changing levels of contrast and intensity by using it to extract target images from one hour of TV recordings of airborne FLIR imagery. (The test imagery was from the Kreb's data set.) In the test data the target contrast varied from approximately 20 per cent to near 100 per cent and the average background intensity (single frame) varied from 0.25 volts to 0.75 volts (relative to a 1 volt maximum). Feature extraction, for the purpose of detecting man made objects, appears consistent over this range of conditions and we expect that the scene adaptation will result in smaller class variances and therefor in improved detection accuracy. The man made object classifier is undergoing real time evaluation at this time and the results will be reported at the workshop. ("Adaptive Threshold for an Image Recognition System")

IMAGE RECOGNITION SYSTEM SIMULATION

The diagram shows the system structure that Honeywell and Purdue have chosen as the simulation goal. The left and right sides of the diagram are concerned with background and targets respectively. The purpose of the background classifier is to establish a local context for detected potential targets, based on both a prior and current information. The target analysis begins with man made object detection based on segmentation of the image and preliminary classification of the segments. From the current Autothreshold results, we feel that the Autoscreener will be able to perform this function. Previous work has shown that one cannot use the same classifier to recognize both small images and large images. Small images can be classified quite well by statistical pattern recognition methods while large images (images that are large enough to show object structure) do not allow good statistical description. Therefore, we follow the MMO detection by a second screening that sorts the detected objects by the size of the images. This screening step will also estimate the actual dimensions of the object and reject objects that are the wrong size for the mission targets. Classification then proceeds as shown with small images being classified statistically and large images being classified structurally.

We have started work on the secondary screening algorithm using the Autoscreener to extract objects and provide image size measurements. Sorting by image size is a trivial task, but there are some difficulties to be overcome in estimating the true dimensions of the object when the object aspect angle is unknown. There is an additional problem in working with our data base in that altitude and viewing angles, needed to determine range, were not recorded and must be estimated. We will implement the algorithm as a software modification to the Autoscreener for on line testing.

We have made some changes to the inference process used in the Prototype Similarity segmentation method, but the work has not proceeded far enough to report at this time.

DATA BASE CONSIDERATIONS

The purpose of this simulation effort is to investigate the application of image understanding technology to a variety of military tactical airborne missions. Thus there are a number of requirements that the experimental data base must meet. In addition to the constraints on target objects, background types, observing platform and sensor characteristics, there is the fact that the sensor data available during a tactical operation will be a continuing sequence of images that are highly correlated. To include the information gain available from sequential processing it is desirable for the data base to be sequential.

The Kreb's data set is an appropiate, realistic collection of imagery, but it does not include imagery from the tactical FLIR's and the low resolution arrays. Honeywell has initiated an effort to obtain imagery from these types of sensors and we would be interested in talking with other DARPA contractors with similar desires.

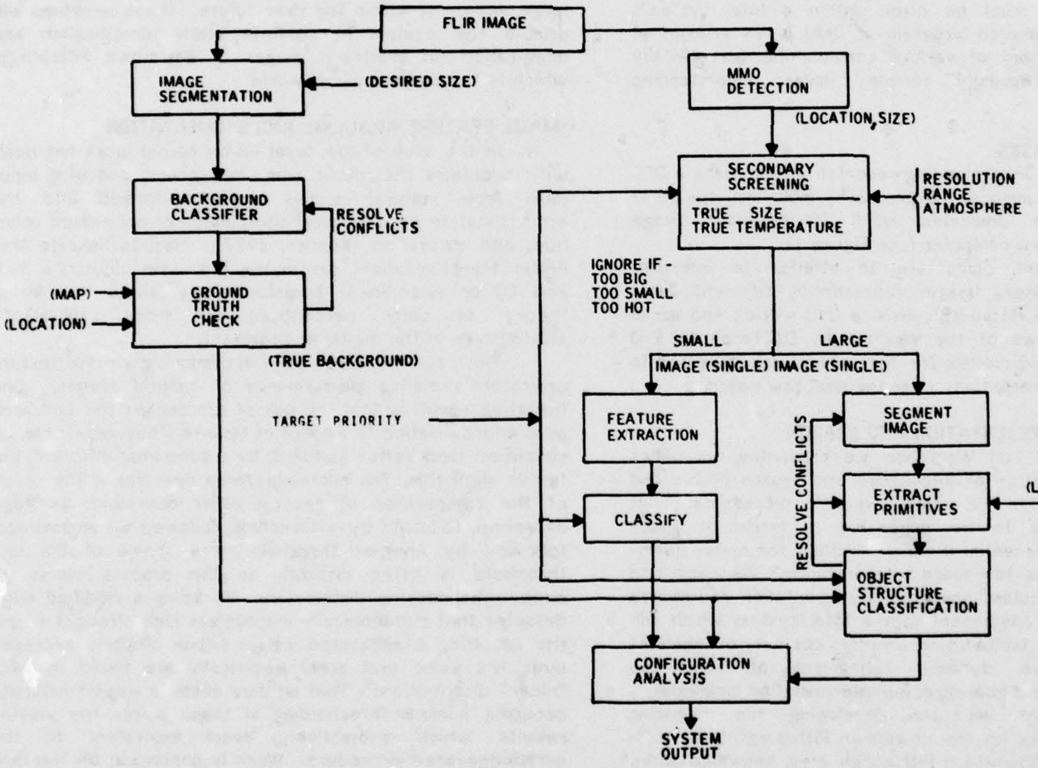


IMAGE UNDERSTANDING RESEARCH AT CMU:
A Progress Report

Raj Reddy
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pa. 15213
September 9, 1977

INTRODUCTION

The primary objective of our research effort is to develop techniques and systems which would lead to successful demonstration of image understanding concepts over a wide variety of tasks, using all the available sources of knowledge. This requires the determination of the type and nature of knowledge that might be applicable in a given task situation. The representation, use, and evaluation of such knowledge must be made within a total system's context. The research program at CMU is an attempt at parallel development of various components, incrementally leading to increasingly complex image understanding systems.

SYSTEMS AND TASKS

The image understanding research at CMU uses a DEC System 10/80, C.mmp (a 16 processor multi-mini computer system), and a dedicated MIPS (Multi-sensor Image Processing System) computer (see Figure 1).

Our present plans are to attempt to interpret uncontrived arbitrary images representing different views of the downtown Pittsburgh area (a 3-D world), and aerial and satellite views of the Washington, D.C. area (a 2-D world). The world models for these tasks are expected to be generated incrementally over the next few years.

KNOWLEDGE REPRESENTATION AND SEARCH

During the last Workshop we presented our views about representation of knowledge and search (Rubin and Reddy, 1977). The PPE graph structure representation of knowledge tends to be expensive in terms of space required, but is essential if we wish to use the faster beam-search techniques for image interpretation. We expect to embed this particular knowledge representation and search as the principal component into a total system which will involve planning (solution in simpler, coarser, or abstract spaces), iterative dynamic refinement of knowledge representation, and goal-directed interpretation strategies.

At present we are developing the following knowledge sources for the downtown Pittsburgh task: a 3-D model of the downtown Pittsburgh area, knowldge about building structures and textures, knowledge about local refinements given coarse recognition (e.g., detecting cars in roads and trees and bushes next to roads), knowledge about shadows occlusions and highlights, and so on. Given our basic approach of iterative refinement of knowledge, we will start with simple versions of these knowledge sources, and refine them as we observe their limitations when applied to different scenes.

Since the last Workshop our work has continued on the ARGOS Image Understanding System (Rubin and Reddy, 1977). Two techniques have been developed for pruning the network as the image is labeled. These techniques enable ARGOS to work with very large networks while maintaining low time and space usage. The first and most powerful pruning method is the implementation of "best lists". Best lists, which are used in the HARPY speech understanding system, are lists of optimal nodes at each step in the network path. In effect, best lists define the "beam" of the search. By limiting the size of the best lists to 20, the system is able to save both search time and state storage. The second pruning technique that has been implemented is quality thresholding. By restricting the best list to those nodes whose likelihoods are above a given threshold, search time is reduced without noticeable loss of labeling accuracy.

ARGOS has also benefited from the addition of a new texture operator called contrast which is derived from the 4th moment. The low-level system now consists of this heavy-texture operator and a low-texture operator which is derived from the median. Each of these operators is applied to the red, green, and blue bands, yielding a feature vector with 6 components.

It is expected that ARGOS can start working with very large networks within the near future. These networks will enable the system to perform scene identification and orientation on arbitrary images of downtown Pittsburgh, which is the current micro-world.

IMAGE FEATURE ANALYSIS AND SEGMENTATION

In the area of low level vision recent work has dealt with problems that occur when red, green, and blue input data from natural scenes are transformed into the approximately psychological coordinates of normalized color, hue, and saturation (Kender, 1977). Results indicate that linear transformations (as in the television industry's Y, I, and Q) or near-linear transformations (as in the Hering theory of color perception) are more satisfactory alternatives in the digital environment.

Work also has been done in comparing various texture operators' relative performance on natural scenes. One tentative result is that the use of microedges per unit area as a approximation to amount of texture ("busyness") can be simplified (and better justified) by a somewhat different, but faster algorithm. The microedge/area operator is the result of the composition of several other operators: an edge detection, followed by a threshold, followed by an average, followed by another threshold. The choice of the last threshold is often difficult, as the process yields an exponential-looking distribution. By using a modified edge detector that monotonically emphasizes high strength edges, the resulting exaggerated values (when similarly averaged over the same unit area) empirically are found to have "nicer" distributions. That is, they seem to exhibit naturally occuring minima; thresholding at these points has yielded results which subjectively seem equivalent to the microedge/area procedure. Work is continuing on this and other texture transforms and measures.

CHANGE DETECTION

We plan to continue experiments in symbolic registration and change detection. As changes due to perspective and scale become more and more dominant, it becomes desirable to view the problem of registration as one of search involving constraint satisfaction based on spatial relationships. We think the model presented in Rubin and Reddy (1977) would also be useful in this case. The annual progress report by CDC (available at this workshop) describes the progress to date on the cooperative image registration research.

IMAGE DATABASE

If we are to have adequate performance and error analysis tools and tools for knowledge source generation, it is desirable to manually (or interactively) generate symbolic descriptions of the images to be analyzed. This and other considerations have led us to begin to develop a unified symbolic and signal image database system. The structure of this database is described in McKeown and Reddy (1977a). We have concentrated our efforts on the generation of hand-segmented and labeled scenes of downtown Pittsburgh. We have twenty such pictures and they are currently the working set for the ARGOS system. A major application of the MIDAS sensor database to the area of performance evaluation of image understanding systems is described in McKeown and Reddy (1977b in this workshop).

ARCHITECTURES FOR IMAGE PROCESSING

It is estimated that we will need processing power of the order of 1 to 10 billion instructions per second in digital image processing systems, requiring rapid response times. We are attempting to develop (in cooperation with CDC) new problem-oriented high speed digital processor architectures for image processing. Given that C.mmp and MIPS are closely coupled multiprocessing systems, we are exploring issues of algorithm decomposition and parallel-pipeline system structures for image processing. Another aspect under study is the development of a special instruction set for image processing using the writable control store available with the PDP-11 processors on C.mmp and MIPS.

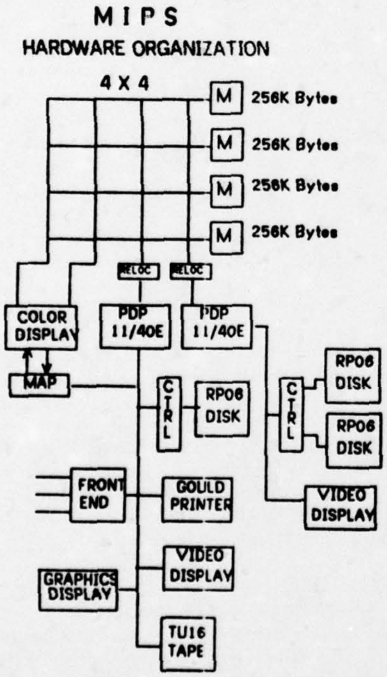


FIGURE 1

Figure 1 shows the hardware organization for our Multi-sensor Image Processing System (MIPS). The basic structure is organized around a 4X4 crosspoint memory switch, a prototype for the 16x16 switch used on C.mmp (Wulf and Bell, 1972). Each of the four memories may be accessed through any of four ports on an arbitrated real-time basis. The bandwidth for each port is about 30 megabits/sec. This design allows concurrent use of memory by four processors. Currently PDP 11/40E processors (Fuller et al., 1976) equipped with writeable control store are connected to two of the ports. The writeable microstore permits the implementation of special image processing instructions. The remaining ports are dedicated to a high bandwidth (about 50 megabits/sec) raster scan color display system. This system displays up to a maximum of 657 pixels by 488 lines (NTSC standard color video) with programmable intensity resolution between 1 and 8 bits per pixel. A color map allows an 8 bit color code to be mapped into 3 ten bit color fields for pseudo-color generation, gamma correction, and data compression. Peripherals such as a 9 track magnetic tape drive, Gould printer and graphics display processor are available as well as three large disk structures totaling 600 megabytes of online storage.

There is 1 megabyte of memory which is organized into four partitions accessable through four ports controlled by a fast crosspoint switch. The switch allows concurrent processing by display hardware and the multi-processors. For example, this organization allows an image frame buffer to be displayed from memory while processors change pixels in the frame. The memory size was chosen so that two images can be held in core simultaneously. This eliminates possible waiting for data by the processors, since bandwidth considerations between disk and memory preclude fast massive transfer of data. Three 200 megabyte disk storage structures provide space for a large online database for image analysis and interpretation. The disk structures are connected to memory by two independant channel controllers which permit a transfer rate of 16 megabits/sec.

Much of our time has been spent since the last Workshop bringing up the UNIX operating system on the MIPS machine. Currently a single processor system with full memory and 400 megabytes of online storage is operational. Communication links are available to a front end terminal concentrator allowing users to communicate from a variety of locations and terminals. A picture processing package (PICPAC) is being implemented in the C language and will provide uniform picture access and display functions.

KNOWLEDGE ACQUISITION

Given the paucity of ideas about type and nature of knowledge used in visual perception, we are continuing our protocol analysis studies in human visual perception. Studies in progress include picture puzzles (Akin and Reddy, 1977), perception as a function of distance, perception in the presence of contradiction, and peep-hole perception studies.

A major dimension used in processing information selectively is stimulus resolution. The overall structure and the fine detail found in the visual environment can be used separately or together to understand its different aspects. In order to understand the use of detail in visual perception an experimental paradigm has been devised. The subjects are required to examine the different size projections of slides of natural scenes. They verbally report what they see during the experiments. The projected images are looked at from 55 feet with six image sizes ranging from 3-1/2" by 5" to 19" by 27".

Initial results indicate that the lack of detailed information in the smaller sizes distorted the scenes beyond recognition (Akin et. al., 1977). The minimum size at which each scene was correctly perceived correlated with the "commonplace"-ness of the overall structure of each scene. Scenes with the sky-buildings-river, person-with-objects-in-the-background structure fit well the image structures ordinarily expected by the subjects. On the other hand, in scenes where the camera was positioned overhead, the structure of the scenes were not recognized in any of the lower three levels. At the lowest levels, scene properties such as edges, textures, and color which were below threshold values of resolution with respect to the size of the receptors in the retina, were perceived erroneously.

CONCLUSION

The representation, use, and evaluation of knowledge within an image understanding system requires parallel development of various compomemts. The research program at CMU represents an attempt to study several different facets of the image understanding problem in a specific problem context, i.e. the 3-D Downtown Pittsburgh task.

REFERENCES

Akin, O. and Reddy, D. R. (1977). "Knowledge Acquisition for Image Understanding," to appear in *Journal of Computer Graphics and Image Processing*, 1977.

Akin, O., McBride, S. and Reddy, R. (1977). "Perception of Visual Detail," Technical Report (in preparation), Dept. of Comp Sci, Carnegie-Mellon University, Pittsburgh, PA.

Fuller, S. H. et al. (1976). "PDP-11E Microprogramming Reference Manual," Technical Report, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA., January, 1976.

Kender, J. (1977). "Instabilities in Color Transformations" in Proceedings of *IEEE Computer Conference on Pattern Recognition and Image Processing* June 1977.

McKeown, D. M. and Reddy, D. R. (1977a). "A Hierarchical Symbolic Representation for an Image Database," Proceeding of *IEEE Workshop on Picture Data Description and Management*, April, 1977.

McKeown, D. M. and Reddy, D. R. (1977b). "The MIDAS Sensor Database and Its Use in Performance Evaluation", in this volumn.

Rubin, S. M. and Reddy, R. (1977). "The LOCUS Model of Search and its Use in Image Interpretation" in Proceedings of *5th IJCAI*, August, 1977

Wulf, W. A. and Bell, C. B. (1972). "C.mmp - a Multi-Mini-Processor", in Proceedings of Fall Joint Computer Conference 1972, Pg. 765-777.

ALGORITHMS AND HARDWARE TECHNOLOGY FOR IMAGE RECOGNITION

Project Status Report - September, 1977

Computer Science Center, University of Maryland, College Park, MD 20742

ABSTRACT

This report summarizes progress made during the period March-September, 1977 in the research being conducted under Contract DAAG53-76C-0138 (DARPA Order 3206). This project is devoted to the development of algorithms for automatic target cueing in FLIR imagery, and to the design of CCD circuits that implement these algorithms. It is a joint effort of the Computer Vision Laboratory at the University of Maryland (Principal Investigators: Profs. David L. Milgram and Azriel Rosenfeld) and the Westinghouse Defense and Electronics Systems Center, Systems Development Division, Baltimore, MD (Program Director: Dr. Glenn E. Tisdale). The project is being monitored by Mr. John Dehne and Dr. George Jones of the U. S. Army Night Vision Laboratory, Ft. Belvoir, VA.

INTRODUCTION

The project reviewed in this status report was initiated on May 1, 1976. Accomplishments during the first ten months were summarized in the previous status report (as of March, 1977), which was included in the Proceedings of the April 1977 Image Understanding Workshop [1]. Further information on this work can be found in the Second Semi-Annual Report on the project [2], covering the period 1 November 1976 - 30 April 1977.

IMAGE MODELING

Two studies have been conducted to analyze the outputs of edge detection and thresholding operators applied to an image region. These studies are of interest in connection with estimating the false alarm rates associated with image segmentation.

In the first study [3], the input image is treated as a stationary random field from a context-independent ensemble. A statistical analysis of the responses of various edge detection operators, including gradients and the Laplacian, to such an image has been conducted. Various stochastic properties of the outputs were predicted, and the results compared with outputs obtained from real and synthetic images.

The second study [4] investigated the results of thresholding a noise image, modelled as a two-dimensional random process completely characterized by its mean and power spectrum. A statistical analysis of the thresholded output has been carried out, and various properties of this output have been derived. Comparisons have been made with the results of thresholding noise images that have been smoothed to varying degrees.

OBJECT EXTRACTION

Several different approaches to extracting objects from a FLIR image have been investigated. One class of approaches is based on thresholding the image using thresholds derived from (gray level, edge strength) scatter diagrams. These thresholds are usually quite good for extracting single objects, but a more refined approach is needed to handle multiple objects. One such approach makes use of scatter diagrams based on the local maxima of edge strength, rather than on all the edge strength values; it is described in greater detail in [5].

A second general approach to object extraction is based on detecting coincidences between the borders of above-threshold connected components and the points of high edge value. An implementation of this approach, called "Superslice", has yielded good segmentations of FLIR windows. Several improvements have also been investigated, e.g., a technique which takes into account how well the high-edge-value points surround the connected region. Further details on this technique can be found in [5].

The Superslice technique can also be used as part of an iterative threshold selection scheme. Specifically, one can histogram the given image; pick a threshold; use Superslice to extract components whose borders have high edge values; discard these components, rehistogram, and repeat the process. Experiments with this approach are also described in [5].

Studies of optimal edge detection

techniques have also been conducted. In particular, the analysis underlying the Hueckel edge detection scheme has been reexamined, and some modifications in this scheme have been made. A class of operators analogous to the Hueckel operator has also been defined. This work will be described in two forthcoming technical reports.

REGION ANALYSIS AND TRACKING

The algorithms for analyzing the connected component structure of extracted image segments have been closely examined in order to clarify various problems associated with their CCD implementation. In particular, a one-pass algorithm has been developed that constructs a tree structure for a thresholded image in which nodes correspond to connected components of object or background and in which the parent relation is based on region enclosure. In addition, the algorithm labels each region, computes a set of features for it, and computes the chain code of its outer boundary. The details of this algorithm can be found in [6].

The objects contained in a sequence of images can be tracked from frame to frame by defining a comparison function that evaluates differences between descriptions of object regions. One can then apply dynamic programming to discover the most temporally consistent region. This region can then be removed from all frames, and the process can be repeated. This approach has been successfully applied to the small sequence data base that is currently available. The algorithm and test results are described in [7].

CIRCUIT DESIGN AND IMPLEMENTATION

Westinghouse has continued to study the CCD focal plane implementation of the algorithms developed on the project. In particular, much effort has been devoted to designing implementations of the connected component labeling process, as described in [8].

Westinghouse has also implemented a circuit that functions as a sorter. This circuit can be used as a basis for implementing histogramming and median filtering operations. A description of it can be found in [9].

REFERENCES

1. Algorithms and Hardware Technology for Image Recognition: Project Status Report - March, 1977, Proceedings: Image Understanding Workshop, DARPA/IPTO, April 1977, pp. 98-100.
2. Algorithms and Hardware Technology for Image Recognition: Second Semi-Annual Report, 1 November 1976 - 30 April 1977, Computer Science Center, University of Maryland, College Park, MD.
3. Durga P. Panda, Statistical Analysis of Some Edge Operators, University of Maryland Computer Science Center Technical Report 558, July 1977.
4. Durga P. Panda, Statistical Properties of Thresholded Images, University of Maryland Computer Science Center Technical Report 559, July 1977.
5. David L. Milgram, Progress Report on Segmentation Using Convergent Evidence, Proceedings: Image Understanding Workshop, DARPA/IPTO October 1977.
6. David L. Milgram, Constructing Trees for Region Description, University of Maryland Computer Science Center Technical Report 541, May 1977.
7. David L. Milgram, Region Tracking Using Dynamic Programming, University of Maryland Computer Science Center Technical Report 539, May 1977.
8. T. J. Willett, CCD Implementation of an Image Segmentation Algorithm, Proceedings: Image Understanding Workshop, DARPA/IPTO, October 1977.
9. T. Schutt, G. Borsuk, and T. J. Willett, A CCD Histogram Sorter: Feasibility Chip, Proceedings: Image Understanding Workshop, DARPA/IPTO, October 1977.

MIT PROGRESS IN UNDERSTANDING IMAGES

Patrick H. Winston

The Artificial Intelligence Laboratory
Massachusetts Institute of Technology

Representation has been treated as the key issue in MIT image understanding work. Horn has used the reflectance map representation to make synthetic images and register aerial photographs with terrain models. Marr has used the primal sketch, the 2 1/2 D sketch, and generalized cones to work toward a comprehensive theory of recognition. Several theses in these various directions have just been completed.

Registering Images

Image understanding must start with the image. We have therefore devoted considerable attention to understanding the image formation process and to exploiting the constraints involved. Image registration, in particular, has been a focus, since putting a new image into correspondence with map coordinates is a certain prerequisite to further processing.

Image registration can be approached in a number of ways. One method requires the discovery and use of very prominent features. Another involves correlation of the image against a synthetic image made from a digital terrain model using the reflectance map to determine the correct intensities from combinations of surface slope, surface material, and sun position.

Horn has shown that the correlation method produces outstanding results, potentially yielding registration accuracy in the subpixel range. Working with his student, Brett Bachman, he devised a method for coping with the computational problems that at first seem to make correlation unacceptably slow. Basically the method involves appproximating registration first with low resolution real and synthetic images before going on to high resolution and high accuracy. At every stage, this keeps the hill-climbing space reasonably free of distracting local maxima. Details can be found in Horn's paper elsewhere in this collection.

Synthetic Images

The synthetic images generated as part of the registration processes have many other uses. Making shaded relief maps is one application, of course, and we have made a number from both high-altitude and low-altitude points of view. Horn's paper, just mentioned, has samples of the high-altitude variety.

Making the low-altitude images involves some complications because switching to an oblique viewing angle introduces a perspective and an interpolation problem. Another student, Tom Strat, has worked on this problem with Horn, testing a variety of algorithms for speed and quality of the resulting product. Work continues to improve the resolution that can be achieved with reasonable computing time.

Multiple Sun Maps

Horn has just begun another map-making project that involves the use of two or more imaginary suns distributed at key points around the sky. Such maps give a faster, better appreciation of the terrain than ordinary shaded maps. More intuitively, it is clear that north-facing slopes will be blue if the imaginary sun to the north is the blue one.

Formally, the reason is that the intensity of a point in an ordinary black-and-white image is not sufficient to determine the surface normal of the surface corresponding to that point. There is constraint, however, and one black-and-white intensity does limit the normal to lie on a definite curve in the reflectance map. Using multiple suns, each of a different color in a different part of the sky, two or more separate curves in reflectance-map space are obtained. Their intersection gives the surface normal unambiguously.

Soon we hope to combine two slope-indicating colors with a third indicating altitude in order to pack in still more information.

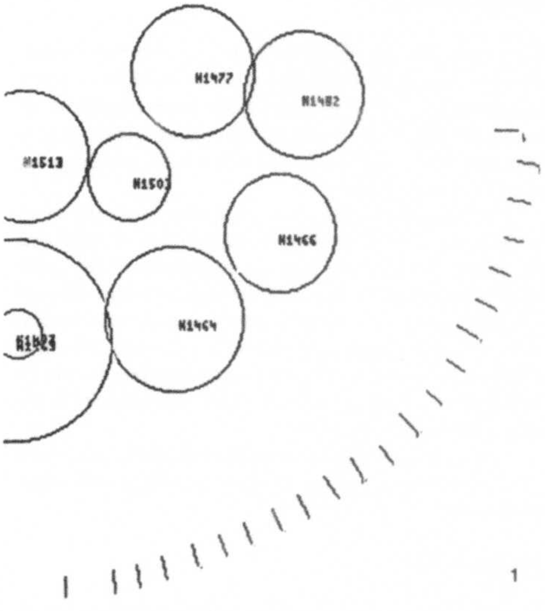
The Albedo Map

Once registration is under control, several related things can be done starting with the same reflectance-map based technology. Change detection and ground cover analysis are two things to which we will be devoting considerable attention. We believe that the ratio of real image intensity to synthetic image intensity will be a good index to ground cover. This ratio does not depend much on sun position, unlike other measures used up to now. We call an image made up of these ratios an albedo map.

Motion

Two theses were completed during the summer having to do with motion understanding. The first, by Shimon Ullman, produced a number of results: the first involved the demonstration that five points on a rigid object seen in three views are enough to determine the relative three-dimensional position of the five points in each of the views; the second involved facts having to do with matching features in one view to those in another view made a short time later. Importantly, a family of experiments demonstrated that the feature correspondence is carried out before features are grouped together into objects. This is an existence proof that a machine can get at distance information using motion without first forming objects and without the obstacles that this involves.

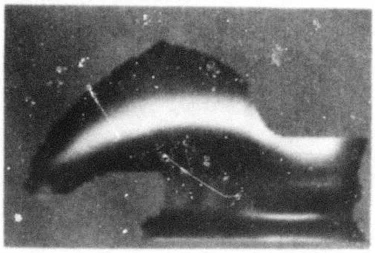
The second thesis on motion vision, by Mark Lavin, investigated the problem of low-level navigation in hilly terrain. Lavin's input consisted of line drawings of hills taken by a moving vehicle whose position, speed, and orientation change. The output, as shown in figure 1 is a map showing the location of the hills observed and the position of the vehicle at each point a snapshot is taken. To do this, Lavin combined Ullman's mathematical results with his own hill-matching program. Extensive error analysis led to accurate algorithms.



1

Shape From Shading

Bob Woodham completed a thesis that was directed at the theoretical problem of getting surface-orientation information from intensity and at the practical problem of analyzing metal castings for defects. On the theoretical side, he was able to exhibit a range-constriction algorithm that does a shape analysis without the integrations required by Horn's previous methods. On the application side, he applied Marr's primal sketch operators to the problem of determining casting grain size and his programs equaled human performance on flat surfaces. As with understanding aerial photographs, part of the problem is registering real images with models. Figure 2 shows a synthetic image of a cast shuttlecock made in preparation for such registration. Note the specular highlight.



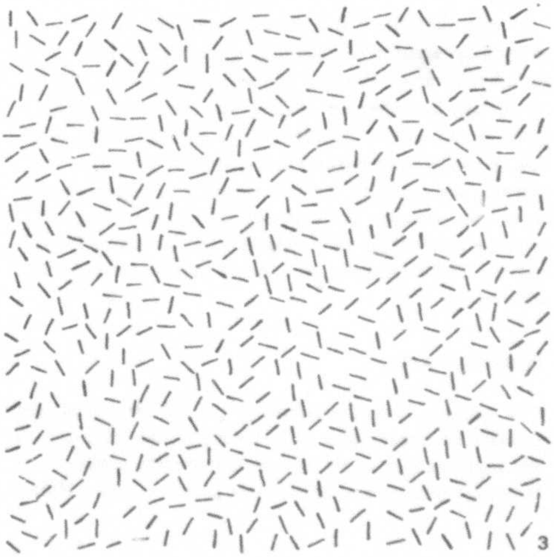
2

Texture

The information in Marr's primal sketch seems to have a major role in determining texture. This is important because texture gradients help build the 2 1/2 D sketch and because classifying textures and discriminating among them is important by itself.

Bruce Schatz has finished a thesis that argues that texture is determined by first order statistics on a subset of the primal sketch descriptors. It seems that only ungrouped edge fragments and virtual lines connecting neighboring edge points are needed.

Moreover, it seems that the analysis of these texture-determining descriptors can be quite coarse. Mike Riley has shown that histograms of line orientation with only five or six buckets seem sufficient for handling the line-orientation part of texture discrimination. Figure 3 shows one of Riley's experimental images. The figure consists mainly of a large square in which the line segments are at random orientations. Inside it, there is a subsquare of about one fourth the size in which the line segments have only three orientations. Curiously, the subsquare is not readily discernable, thus arguing for the hypothesis that the texture discrimination machinery in humans is not very refined.



3

This is encouraging since it suggests machines can do well without exorbitant computation. Related experiments seem to show that intensity histograms can be quite coarse as well, although work in this direction is preliminary.

Representation

We are committed to the idea that good representation is the key to successful image understanding. The reflectance map and the albedo map are examples of good representations oriented toward understanding and exploiting shading. The primal sketch, the 2 1/2 D sketch, and the generalized cones invented by Binford at Stanford University are examples oriented toward *recognition*.

Each of these representations was devised to make some particular kind of information explicit. Each in turn helps to define the computational problems that eventually lead to working algorithms.

With the basic work on the primal sketch complete, attention has been turned to the 2 1/2 D sketch and to the question of making it from information in the primal sketch. Our current design calls for representation of depth and surface orientation as well as contours of discontinuity in these quantities. Additional internal computational structure will maintain consistency between them all.

At the generalized-cone level of representation, Marr and Keith Nishihara continue to work out criteria for applicability and means by which hierarchies of descriptions can be assembled together using body-centered coordinate systems.

One seemingly important theorem discovered by Marr states the following: given some simple continuity assumptions, if the points on a surface that correspond to the observed boundary of that surface all lie in a plane, then the surface is part of a generalized cone.

References

Kenneth Forbus, "Light Source Effects," AIM-422, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

Berthold K. P. Horn and Brett L. Bachman, "Using Synthetic Images to Register Real Images with Surface Models," AIM-437, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

Mark A. Lavin, "Computer Analysis of Scenes from a Moving Viewing Point," Ph.D. Thesis, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

David Marr, "Representing Visual Information," AIM-415, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

Michael D. Riley, "Discrimination of Bar Textures with Differing Orientation and Length Distributions," B.S. Thesis, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

Bruce R. Schatz, "The Computation of Immediate Texture Discrimination," M.S. Thesis, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

Thomas M. Strat, "Automatic Production of Shaded Orthographic Projections of Terrain," B.S. Thesis, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

Shimon Ullman, "The Interpretation of Visual Motion," Ph.D. Thesis, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.

Robert J. Woodham, "Reflectance Map Techniques for Analyzing Surface Defects in Metal Castings," Ph.D. Thesis, The Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1977.